Universidad de Deusto

Facultad de Ingeniería

Doctorado en Ingeniería Informática y Telecomunicación

Estrategia de Enfriamiento Adaptativo para el Algoritmo del Recocido Simulado

Jesús Manuel Riaño Sierra

2015

Bilbao

Universidad de Deusto

Facultad de Ingeniería

Doctorado en Ingeniería Informática y Telecomunicación

Estrategia de Enfriamiento Adaptativo para el Algoritmo del Recocido Simulado

Tesis doctoral presentada por D. Jesús Manuel Riaño Sierra

Dirigida por el Dr. José Fernando Díaz Martín

El Director

El Doctorando

Bilbao, a 29 de julio de 2015

A Izaskun,

Carmen y Estibaliz.

ABSTRACT

Simulated Annealing (SA) is one of the most important present meta-heuristics or general algorithms of combinatorial optimization. In spite of its high computational cost, its features of convergence towards high quality solutions are well known. Thus, numerous research works on the convergence acceleration of the algorithm especially concerning the treatment of the temperature parameter have been produced, by means of what is known as chilling programs or strategies. This Thesis aims at designing an adaptive chilling strategy for the algorithm of Simulated Annealing in order to reduce its computational cost, but respecting the quality of the solutions. In this sense a comparative study of the performance of the Simulated Annealing has been produced including the traditional proposal by Kirkpatrick, Gelatt and Vecchi [KIRK 83], as well as more recent ones. The new proposed chilling strategy is based on the dynamic optimal adjustment of two parameters of the algorithm: the initial value of the temperature and the temperature decrease factor after every chilling cycle. Two traditional problems of combinatorial optimization have been used: the Traveling Salesman Problem (TSP) and the Quadratic Assignment Problem (QAP), analysing the performance of the chilling strategy related to both the time efficiency and the quality of the solutions. In addition, a business field optimization problem is proposed as an example of the practical application of the new algorithm: The optimal selection of investment portfolios.

Keywords: Combinatorial Optimization, Simulated Annealing, Chilling Programs, Investment Portfolios Optimization.

RESUMEN

El Recocido Simulado es una de las más importantes meta-heurísticas o algoritmos generales de optimización combinatoria existentes, siendo bien conocidas sus propiedades de convergencia hacia soluciones de alta calidad, aunque con un elevado coste computacional. Esto ha dado origen a numerosos trabajos de investigación sobre aceleración de la convergencia del algoritmo, especialmente en el tratamiento del parámetro de temperatura, mediante lo que se conoce como programas o estrategias de enfriamiento. Esta tesis tiene como objetivo diseñar una estrategia de enfriamiento adaptativo para el algoritmo del recocido simulado con el fin de reducir su coste computacional manteniendo la calidad de las soluciones. Para ello, en primer lugar se ha elaborado un estudio comparativo del comportamiento del recocido simulado respecto a varias estrategias de enfriamiento distintas, incluyendo la clásica propuesta por Kirkpatrick, Gelatt y Vecchi [KIRK83], así como otras más recientes. La nueva estrategia de enfriamiento propuesta se basa en el ajuste dinámico óptimo de dos parámetros de control del algoritmo: el valor inicial de la temperatura y el factor de decremento de la temperatura tras cada ciclo de enfriamiento. En el análisis práctico del nuevo algoritmo se usan dos problemas clásicos de optimización combinatoria: el problema del viajante comercial y el problema de la asignación cuadrática, evaluando el comportamiento de la estrategia de enfriamiento tanto respecto a la eficiencia temporal, como a la calidad de las soluciones. Adicionalmente, se propone como ejemplo de aplicación práctica del nuevo algoritmo un problema de optimización de ámbito empresarial: La selección óptima de carteras de inversión.

Palabras clave: Optimización Combinatoria, Recocido Simulado, Programas de Enfriamiento, Optimización de Carteras de Inversión.

LABURPENA

Simulatutako suberaketa meta-heuristika edo konbinazio optimizazio algoritmo orokor garrantzitsuenetakoa da eta kalitate goreneko soluzioak ahalbidetzen dituzten bere konbergentzia ezaugarriak aski ezagunak dira kostu konputazional handia izan arren. Honek algoritmoaren konbergentzia azelerazioari buruzko ikerketa ugari sorrarazi ditu, batez ere, tenperaturaren parametroaren tratamendu arloan, hozte estrategia edo programa izenaz ezaguna dena. Tesi honen helburua simulatutako suberaketaren algoritmorako hozte moldatzaile estrategia diseinatzea da, bere koste konputazionala murrizteaz gain, soluzioen kalitatea mantentzea lortzen dela. Horretarako, lehenik eta behin, simulatutako suberaketaren portaeraren konparazio ikerketa egin da hozte estrategia ezberdinekin alderatuta, Kirkpatrick, Gelatt eta Vechi (KIRK83) eta beste berriago batzuk barne daudela. Proposatutako hozte estrategia berria algoritmoaren kontrolaren parametro biren doikuntza dinamiko hoberenean oinarritzen da: tenperaturaren hasierako balioa eta hozte ziklo bakoitzaren ondorengo tenperaturaren gutxitze faktorea: Algoritmo berriaren analisi praktikoan konbinazio optimizazioaren gaineko problema klasiko bi erabiltzen dira: merkataritza bidaiariaren problema eta esleipen koadratikoaren problema, bietan hozte estrategiaren portaera aztertzen da denbora efizientzia eta soluzioen kalitatea kontuan hartuta. Horrez gain, enpresa arloko optimizazio problema bat proposatzen da algoritmo berriaren aplikazio praktiko bezala: inbertsio zorroen hautapen hoberena.

Hitz gakoak: konbinazio optimizazioa, simulatutako suberaketa, hozte programak, inbertsio zorroen optimizazioa.

Agradecimientos

Esta tesis es el resultado de una gran dedicación personal, un largo camino desde su inicio y definición hasta este momento en el que finaliza. Un trabajo que no hubiera sido posible sin la colaboración y el apoyo que me han prestado muchas personas durante estos años. Amigos que se han preocupado y me han acompañado durante todo este tiempo y que me gustaría agradecer públicamente su apoyo y su cariño.

Principalmente, quiero agradecer a mi director de tesis, Fernando Díaz, su labor de orientación sobre el tema de la tesis doctoral, así como el seguimiento realizado durante su desarrollo, junto a sus consejos y trabajo aportados para la definición de la misma.

Agradecer a Javier Cuñado, Aitor Pérez, Belén Gandarias, Roberto Carballedo, Aitor Simón, Juan Carlos Duque, Iker Jamardo, Susana Martínez y, por supuesto, a todos mis compañeros del departamento por la ayuda y la amistad que me han prestado durante todo este tiempo.

Siempre son de agradecer las palabras de ánimo, los consejos, las aportaciones y el interés de todos los amigos que me han acompañado en este largo camino: José Vicente Ugarte, Tontxu Campos, José Luis del Val, Cristina Diago, Javier López Arístegui y, de manera muy especial, Roberto San Salvador del Valle que me ha guiado en mi carrera profesional.

A mi hermano Javier, por haberme proporcionado los conocimientos sobre el problema de la selección de carteras de inversión y todo el tiempo que ha dedicado para orientarme.

También quiero dar las gracias a toda mi Familia, por su apoyo, cariño y compromiso que han mostrado en mi decisión de realizar esta tesis.

A mi amigo Fernando, por su ayuda y comprensión en los momentos difíciles habidos en estos años de trabajo. Y por último, con todo mi cariño, agradecer a mi mujer Izaskun su apoyo, su generosidad, su ánimo y su compañía, gracias a los cuales he podido finalizar la investigación de esta tesis.

ÍNDICE

ÍNDICE i

ÍNDICE DE TABLAS v

ÍNDICE DE FIGURAS xv

1. INTRODUCCIÓN 1

1.1. Planteamiento del problema 3

1.2. Descripción general de la estrategia de enfriamiento adaptativo 8

1.3. Principales aportaciones 9

1.4. Esquema de los restantes capítulos 11

2. OPTIMIZACIÓN COMBINATORIA 15

2.1. Optimización combinatoria: Conceptos básicos 17

2.1.1. Introducción a la optimización combinatoria 17

2.1.2. Soluciones de un problema de optimización combinatoria 20

2.1.3. Complejidad algorítmica 25

2.1.4. Tipos notables de problemas de optimización combinatoria 31

2.1.5 Resolución exacta de los problemas de optimización combinatoria: Búsqueda exhaustiva 49

2.1.6. Resolución aproximada de los problemas de optimización combinatoria: Heurísticas y meta-heurísticas 51

2.2. Algoritmos transformativos de búsqueda simple I: Búsqueda local 61

2.2.1. Búsqueda local 61

2.2.2. Búsqueda de vecindad variable 63

2.2.3. Búsqueda local iterativa 66

2.3. Algoritmos transformativos de búsqueda simple II: Recocido simulado 70
2.4. Algoritmos transformativos de búsqueda simple III: Búsqueda tabú 74
2.5. Algoritmos transformativos poblacionales I: Algoritmos evolutivos 78
2.5.1. Algoritmos evolutivos: Conceptos básicos 78
2.5.2. Algoritmos genéticos 79
2.5.3. Algoritmos meméticos 82
2.5.4. Algoritmos culturales 83
2.6. Algoritmos transformativos poblacionales II: Optimización por inteligencia colectiva 84
2.6.1. Fundamentos de la optimización por inteligencia colectiva 84
2.6.2. Optimización por enjambre de partículas inteligentes 85
2.6.3. Optimización por colonia de abejas artificiales 88
2.7. Algoritmos transformativos poblacionales III: Optimización por combinación de soluciones 90
2.7.1. Fundamentos de la optimización por combinación de soluciones 90
2.7.2. Búsqueda dispersa 91
2.7.3. Reencadenamiento de trayectorias 93
2.8. Algoritmos constructivos I: Algoritmos voraces 94
2.8.1. Algoritmos voraces simples 94
2.8.2. GRASP 95
2.9. Algoritmos constructivos II: Optimización por colonia de hormigas 97

3. RECOCIDO SIMULADO 101
3.1. Introducción al algoritmo recocido simulado 103
3.1.1. Fundamentos físicos del recocido simulado 104
3.1.2. Algoritmo del recocido simulado 106
3.2. Programas de enfriamiento 112

3.2.1. Enfriamiento monótono multiplicativo 113
3.2.2. Enfriamiento monótono aditivo 116
3.2.3. Enfriamiento no monótono adaptativo 119
3.3. Comparativa de programas de enfriamiento del recocido simulado 120
3.3.1. Descripción de los problemas de optimización combinatoria 121
3.3.2. Elección de parámetros 125
3.3.3. Resultados de las pruebas 128
3.3.4. Análisis estadístico 134
3.3.5. Análisis de los resultados 153

4. ENFRIAMIENTO ADAPTATIVO 155
4.1. Influencia de la temperatura inicial en la eficiencia del algoritmo 158
4.1.1. Elección de parámetros de enfriamiento 158
4.1.2. Resultados de las pruebas 161
4.1.3. Análisis de los resultados 165
4.2. Influencia del parámetro α en la eficiencia del algoritmo 179
4.2.1. Elección de parámetros de enfriamiento 179
4.2.2. Resultados de las pruebas 180
4.2.3. Análisis de los resultados 184
4.3. Influencia combinada de la temperatura inicial y el parámetro α en la eficiencia del algoritmo 201
4.3.1. Elección de parámetros de enfriamiento 201
4.3.2. Resultados de las pruebas 202
4.3.3. Análisis de los resultados 208
4.4. Análisis conjunto de las pruebas realizadas 238

5. UNA APLICACIÓN PRÁCTICA DE LA ESTRATEGIA DE ENFRIAMIENTO ADAPTATIVO: PROBLEMA DE SELECCIÓN DE CARTERAS DE INVERSIÓN 241

5.1. El problema de la selección de carteras de inversión 243

5.1.1. Teoría Moderna de Carteras 245

5.1.2. Rendimiento y Riesgo de una Cartera..... 248

5.1.3. Frontera Eficiente 251

5.2. Caso de estudio: Problema de selección de carteras de inversión 252

5.2.1. Técnicas para la Optimización de Carteras..... 253

5.3. Formulación matemática del problema de selección de carteras de inversión..... 256

5.4. Resolución del problema de la cartera de valores mediante recocido simulado con la estrategia de enfriamiento adaptativo óptimo..... 261

6. CONCLUSIONES Y LÍNEAS FUTURAS..... 267

6.1. Conclusiones sobre el trabajo realizado..... 269

6.2. Líneas futuras de investigación..... 274

BIBLIOGRAFÍA.....277

ÍNDICE DE TABLAS

Tabla 2.1: Crecimiento de la sucesión exponencial $a_n = 2^n$ 27

Tabla 2.2: Principales meta-heurísticas de optimización combinatoria..... 59

Tabla 3.1: Parámetros de los programas de enfriamiento 129

Tabla 3.2: Resultados TSP 47 129

Tabla 3.3: Resultados TSP 80 130

Tabla 3.4: Resultados QAP 47 131

Tabla 3.5: Resultados QAP 80 132

Tabla 3.6: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo logarítmico 136

Tabla 3.7: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo logarítmico 137

Tabla 3.8: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo lineal 138

Tabla 3.9: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo lineal 139

Tabla 3.10: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo cuadrático 140

Tabla 3.11: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo cuadrático 141

Tabla 3.12: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo lineal 142

Tabla 3.13: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo lineal..... 143

Tabla 3.14: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo cuadrático 144

Tabla 3.15: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo cuadrático 145

Tabla 3.16: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo exponencial 146

Tabla 3.17: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo exponencial..... 147

Tabla 3.18: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo trigonométrico..... 148

Tabla 3.19: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo trigonométrico 149

Tabla 3.20: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento no monótono adaptativo 150

Tabla 3.21: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento no monótono adaptativo..... 151

Tabla 3.22: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de objetivos según la distribución normal 152

Tabla 3.23: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de número de iteraciones según la distribución normal 152

Tabla 4.1: Valor de T_0 según el nº ciclos que se ejecutan 160

Tabla 4.2: Resultados TSP 47 161

Tabla 4.3: Resultados TSP 80 162

Tabla 4.4: Resultados QAP 47 163

Tabla 4.5: Resultados QAP 80 164

Tabla 4.6: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 90% del total de ciclos y el enfriamiento con temperatura inicial estándar 167

Tabla 4.7: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 80% del total de ciclos y el enfriamiento con temperatura inicial estándar 169

Tabla 4.8: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 70% del total de ciclos y el enfriamiento con temperatura inicial estándar 170

Tabla 4.9: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 60% del total de ciclos y el enfriamiento con temperatura inicial estándar 171

Tabla 4.10: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 50% del total de ciclos y el enfriamiento con temperatura inicial estándar 172

Tabla 4.11: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 40% del total de ciclos y el enfriamiento con temperatura inicial estándar 173

Tabla 4.12: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 30% del total de ciclos y el enfriamiento con temperatura inicial estándar 174

Tabla 4.13: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 20% del total de ciclos y el enfriamiento con temperatura inicial estándar 175

Tabla 4.14: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 10% del total de ciclos y el enfriamiento con temperatura inicial estándar 176

Tabla 4.15: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 5% del total de ciclos y el enfriamiento con temperatura inicial estándar 177

Tabla 4.16: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de objetivos según la distribución normal 178

Tabla 4.17: Resultados TSP 47 180

Tabla 4.18: Resultados TSP 80 181

Tabla 4.19: Resultados QAP 47 182

Tabla 4.20: Resultados QAP 80 183

Tabla 4.21: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 1% y el enfriamiento con parámetro α constante..... 187

Tabla 4.22: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 1% y el enfriamiento con parámetro α constante..... 188

Tabla 4.23: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 2% y el enfriamiento con parámetro α constante..... 189

Tabla 4.24: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 2% y el enfriamiento con parámetro α constante 190

Tabla 4.25: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 3% y el enfriamiento con parámetro α constante 191

Tabla 4.26: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 3% y el enfriamiento con parámetro α constante 192

Tabla 4.27: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 4% y el enfriamiento con parámetro α constante 194

Tabla 4.28: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 4% y el enfriamiento con parámetro α constante 194

Tabla 4.29: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 5% y el enfriamiento con parámetro α constante 195

Tabla 4.30: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 5% y el enfriamiento con parámetro α constante 196

Tabla 4.31: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 10% y el enfriamiento con parámetro α constante 197

Tabla 4.32: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 10% y el enfriamiento con parámetro α constante 198

Tabla 4.33: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de objetivos según la distribución normal 199

Tabla 4.34: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de número de iteraciones según la distribución normal 199

Tabla 4.35: Resultados TSP 47 con los diferentes valores del parámetro α 202

Tabla 4.36: Resultados TSP 80 con los diferentes valores del parámetro α 204

Tabla 4.37: Resultados QAP 47 con los diferentes valores del parámetro α 205

Tabla 4.38: Resultados QAP 80 con los diferentes valores del parámetro α 207

Tabla 4.39: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar 211

Tabla 4.40: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar 212

Tabla 4.41: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar 213

Tabla 4.42: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar 214

Tabla 4.43: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar 215

Tabla 4.44: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar 216

Tabla 4.45: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar 217

Tabla 4.46: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar 218

Tabla 4.47: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar 219

Tabla 4.48: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar 220

Tabla 4.49: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar 221

Tabla 4.50: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar 222

Tabla 4.51: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar 223

Tabla 4.52: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar 224

Tabla 4.53: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar 225

Tabla 4.54: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar 226

Tabla 4.55: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar 227

Tabla 4.56: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar 228

Tabla 4.57: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar 229

Tabla 4.58: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar 230

Tabla 4.59: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar 231

Tabla 4.60: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar 232

Tabla 4.61: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar 233

Tabla 4.62: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar 234

Tabla 4.63: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de objetivos según la distribución normal 235

Tabla 4.64: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de número de iteraciones según la distribución normal 236

Tabla 5.1: Promedios de rentabilidades y desviaciones típicas semanales. 263

Tabla 5.2: Parámetros de los programas de enfriamiento 264

Tabla 5.3: Resultados programas de enfriamiento estándar y adaptativo óptimo 265

ÍNDICE DE FIGURAS

Figura 2.1: Relación entre las clases P, NP, NP-completo, NP-duro 30

Figura 3.1: Curva de enfriamiento multiplicativo exponencial, $T_k = T_0 \cdot \alpha^k$ .. 114

Figura 3.2: Curva de enfriamiento multiplicativo logarítmico, $T_k = \frac{T_0}{1+\alpha \ln(1+k)}$ 114

Figura 3.3: Curva de enfriamiento multiplicativo lineal, $T_k = \frac{T_0}{1+\alpha k}$ 115

Figura 3.4: Curva de enfriamiento multiplicativo cuadrático, $T_k = \frac{T_0}{1+\alpha k^2}$ .. 115

Figura 3.5: Curva de enfriamiento aditivo lineal, $T_k = T_n + (T_0 - T_n)\left(\frac{n-k}{n}\right)$ 116

Figura 3.6: Curva de enfriamiento aditivo cuadrático,

$T_k = T_n + (T_0 - T_n)\left(\frac{n-k}{n}\right)^2$ 117

Figura 3.7: Curva de enfriamiento aditivo exponencial,

$T_k = T_n + (T_0 - T_n)\left(\frac{1}{1+e^{\frac{10}{n}\left(k-\frac{1}{2}n\right)}}\right)$ 118

Figura 3.8: Curva de enfriamiento aditivo trigonométrico,

$T_k = T_n + \frac{1}{2}(T_0 - T_n)\left(1+\cos\left(\frac{k\pi}{n}\right)\right)$ 118

Figura 3.9: Curva de enfriamiento no monótono adaptativo

$T = \left(1+\frac{|f(\boldsymbol{x}) - f^*|}{f(\boldsymbol{x})}\right)T_k$,combinado con el enfriamiento multiplicativo exponencial, $T_k = T_0 \cdot \alpha^k$ 120

Figura 3.10: Comparación de resultados (objetivo) TSP 47 130

Figura 3.11: Comparación de resultados (nº iteraciones) TSP 47 130
Figura 3.12: Comparación de resultados (objetivo) TSP 80 131
Figura 3.13: Comparación de resultados (nº iteraciones) TSP 80 131
Figura 3.14: Comparación de resultados (objetivo) QAP 47 132
Figura 3.15: Comparación de resultados (nº iteraciones) QAP 47 132
Figura 3.16: Comparación de resultados (objetivo) QAP 80 133
Figura 3.17: Comparación de resultados (nº iteraciones) QAP 80 133
Figura 4.1: Comparación de resultados (objetivo) TSP 47 162
Figura 4.2: Comparación de resultados (nº iteraciones) TSP 47 162
Figura 4.3: Comparación de resultados (objetivo) TSP 80 163
Figura 4.4: Comparación de resultados (nº iteraciones) TSP 80 163
Figura 4.5: Comparación de resultados (objetivo) QAP 47 164
Figura 4.6: Comparación de resultados (nº iteraciones) QAP 47 164
Figura 4.7: Comparación de resultados (objetivo) QAP 80 165
Figura 4.8: Comparación de resultados (nº iteraciones) QAP 80 165
Figura 4.9: Comparación de resultados (objetivo) TSP 47 181
Figura 4.10: Comparación de resultados (nº iteraciones) TSP 47 181
Figura 4.11: Comparación de resultados (objetivo) TSP 80 182
Figura 4.12: Comparación de resultados (nº iteraciones) TSP 80 182
Figura 4.13: Comparación de resultados (objetivo) QAP 47 183
Figura 4.14: Comparación de resultados (nº iteraciones) QAP 47 183
Figura 4.15: Comparación de resultados (objetivo) QAP 80 184
Figura 4.16: Comparación de resultados (nº iteraciones) QAP 80 184
Figura 4.17: Comparación de resultados (objetivo) TSP 47 203
Figura 4.18: Comparación de resultados (nº iteraciones) TSP 47 203
Figura 4.19: Comparación de resultados (objetivo) TSP 80 204

Figura 4.20: Comparación de resultados (nº iteraciones) TSP 80……205

Figura 4.21: Comparación de resultados (objetivo) QAP 47 .....……206

Figura 4.22: Comparación de resultados (nº iteraciones) QAP 47 ……206

Figura 4.23: Comparación de resultados (objetivo) QAP 80 .....……207

Figura 4.24: Comparación de resultados (nº iteraciones) QAP 80 ……208

Figura 5.1: Ejemplo de una cartera de inversión..... 245

Figura 5.2: Frontera Eficiente..... 252

Figura 5.3: Cartera de inversión con el programa de enfriamiento estándar (solución óptima) 266

Figura 5.4: Cartera de inversión con el programa de enfriamiento adaptativo óptimo (solución óptima) 266

Capítulo 1: Introducción

"Si lo puedes soñar,
lo puedes lograr"

Walt Disney

1.1. PLANTEAMIENTO DEL PROBLEMA

Una de las cuestiones de mayor interés en las sociedades humanas modernas es la mejora de resultados en el desarrollo de tareas dentro de prácticamente cualquier ámbito (ciencia y tecnología, gestión empresarial, economía, administración, banca, etc.). Este proceso de mejora de resultados se conoce también como proceso de optimización. Desde un punto de vista científico, los procesos de optimización se concretan en la búsqueda de la mejor solución a un problema entre un conjunto de soluciones posibles con un objetivo concreto, optimizar – minimizar o maximizar – una determinada función.

Dentro de la ingeniería puede encontrarse un gran número de problemas relacionados con la optimización, tales como los problemas de planificación de la producción y de la distribución propios de la organización industrial y la logística, el diseño de circuitos electrónicos, el diseño de redes de telecomunicaciones, o el control de procesos industriales. Algunos problemas de optimización de complejidad reducida pueden ser resueltos fácilmente mediante técnicas clásicas de optimización matemática, tales como las técnicas de programación matemática (lineal, entera, no lineal, dinámica, etc.), mientras que otros problemas son difíciles de resolver en la práctica, entendiéndose por problema de optimización *difícil de resolver* aquel para el que no se puede encontrar la solución óptima en un tiempo razonable.

En función de las variables que intervienen en el problema, el estudio de la optimización se divide en optimización continua y optimización discreta o combinatoria.

La *optimización continua* se caracteriza por el estudio y resolución exacta de problemas con o sin restricciones en los que las variables que intervienen son de tipo continuo.

La *optimización combinatoria* tiene por objeto el estudio y la resolución algorítmica de problemas de optimización con o sin restricciones en los que las variables constituyentes son de tipo discreto o finito. La complejidad de los problemas de optimización combinatoria se mide en función del coste computacional del algoritmo que obtiene la solución óptima global del problema, siendo de especial relevancia aquellos que precisan algoritmos de complejidad temporal superior a la potencial (exponencial, factorial o potencial-exponencial) con respecto al tamaño del problema, denominados también problemas NP-completos.

En la práctica, los problemas NP-completos de optimización no pueden resolverse de forma exacta, es decir, asegurando que el resultado obtenido corresponde a la solución óptima global del problema, ya que ello conlleva un tiempo inviable incluso para los más potentes computadores. Por lo tanto, el objetivo en este tipo de problemas es hallar una solución casi-óptima en un tiempo factible (optimización aproximada). Como consecuencia de la necesidad de algoritmos cada vez más eficientes, estos problemas han sido objeto de extenso estudio en las áreas de Matemática Discreta, y de Computación e Inteligencia Artificial, especialmente a partir del desarrollo de la teoría de la NP-completud en los años setenta [GARE79].

La resolución de problemas NP-completos de optimización discreta es un área de gran interés científico en la actualidad, y puede orientarse de dos formas:

- Resolución mediante algoritmos adaptados al problema a resolver, también llamados algoritmos heurísticos o simplemente heurísticas, que usan conocimiento específico del problema para facilitar o acelerar la

búsqueda de la solución óptima o casi-óptima. Los algoritmos heurísticos suelen ser muy potentes y útiles en su aplicación práctica, pero para su diseño se requiere tener un conocimiento muy preciso de las características del problema. Por ello, las heurísticas son comunes en problemas de optimización muy simplificados o de ámbito académico, pero mucho menos frecuentes en problemas del mundo real de complejidad notable.

- Resolución mediante algoritmos generales independientes del problema a resolver, también llamados algoritmos meta-heurísticos o meta-heurísticas. Estos algoritmos se basan en características generales o comunes a todos los problemas de optimización combinatoria, por lo que para su diseño no se precisa del conocimiento específico de los problemas, al menos en lo que se refiere a la técnica concreta de optimización utilizada. Se trata de algoritmos comparativamente menos eficientes que los heurísticos, pero que pueden aplicarse con poco esfuerzo a problemas complejos para los que no existan o no se conozcan heurísticas apropiadas. Obviamente, en su aplicación a problemas concretos de optimización es necesario adaptar de alguna forma el algoritmo meta-heurístico al problema, pero dicha adaptación se reduce normalmente a garantizar la viabilidad de la solución obtenida respecto al conjunto de restricciones del problema, no al método de optimización en que se fundamenta el algoritmo.

Estas dos orientaciones pueden también combinarse para formar algoritmos híbridos en los que bajo una estructura general de resolución o meta-heurística, se utilizan heurísticas específicas de optimización con el objetivo de mejorar la eficiencia del algoritmo en conjunto.

Las meta-heurísticas o técnicas generales de resolución de problemas NP-completos de optimización combinatoria pueden clasificarse en diferentes tipos importantes. En primer lugar, están los algoritmos trayectoriales, en los que se parte de soluciones candidatas del problema, completas y viables respecto a las restricciones, que van modificándose parcialmente según avanza el proceso

de optimización, formando "trayectorias" por el espacio de soluciones del problema. Los algoritmos trayectoriales se subdividen a su vez en algoritmos de búsqueda simple, cuando se considera una única solución candidata que se transforma en el tiempo, y algoritmos de búsqueda múltiple o poblacionales, en los que se contemplan simultáneamente varias soluciones candidatas que evolucionan en el tiempo con una cierta interacción. Entre las meta-heurísticas de búsqueda simple están los algoritmos deterministas de *búsqueda local* (Local Search) [AART97], el algoritmo estocástico del *recocido simulado* (*Simulated Annealing*) [KIRK83], y la *búsqueda tabú* (*Tabu Search*) [GLOV86]. Los principales algoritmos trayectoriales poblacionales son los *algoritmos genéticos* (*Genetic Algorithms*) [HOLL73], [GOLD89], inspirados en la genética y la teoría de la evolución de las especies, los *algoritmos meméticos* (*Memetic Algorithms*) [MOSC99], variante de los algoritmos genéticos que incorpora el denominado aprendizaje cultural propio de las sociedades humanas, así como la optimización por *enjambre de partículas inteligentes* (*Intelligent Particle Swarm Optimization*) [KENN95], basada en el comportamiento de ciertos grupos de animales (bancos de peces, bandadas de pájaros, enjambres de abejas) cuando realizan actividades de migración, búsqueda de alimentos, o defensa colectiva.

En segundo lugar, están los algoritmos constructivos, en los que la solución del problema se "construye" paso a paso asignando secuencialmente valores a las variables de acuerdo con algún criterio de optimización. En cada paso se comprueban las restricciones del problema con respecto a las variables ya asignadas, y si alguna restricción no se cumple se vuelve atrás. Estos algoritmos también se subdividen en algoritmos constructivos simples, cuando el algoritmo construye una única solución del problema en cada ejecución, y algoritmos constructivos múltiples o poblacionales, en los que se construyen simultáneamente varias soluciones con una cierta interacción. Entre las meta-heurísticas constructivas simples están los *algoritmos voraces o golosos* (*Greedy Algorithms*) [CORM01], y el *algoritmo GRASP* (*Greedy Randomized Adaptive Search Procedures*) [FEO95]. Por otra parte, la más conocida de las meta-heurísticas constructivas poblacionales es la *optimización por colonia de hormigas* (*Ant Colony Optimization*) [DORI99].

Como se ha mencionado, el recocido simulado es una de las más importantes meta-heurísticas de búsqueda simple existentes, siendo bien conocidas sus propiedades de convergencia hacia soluciones de alta calidad, aunque con un elevado coste computacional. Esto ha dado origen a numerosos trabajos de investigación sobre aceleración de la convergencia del algoritmo, especialmente en el tratamiento del parámetro de temperatura, mediante lo que se conoce como programas o estrategias de enfriamiento.

La hipótesis de la investigación es la siguiente: Puede determinarse una estrategia de enfriamiento para el recocido simulado que consiga acelerar su convergencia, esto es, reducir el elevado tiempo de ejecución característico de esta meta-heurística, sin que ello suponga un detrimento en la calidad media de la soluciones obtenidas. Para validar esta hipótesis, se realiza un riguroso estudio comparativo del comportamiento del recocido simulado respecto a varias estrategias de enfriamiento distintas, tomando como referencia la clásica propuesta por Kirkpatrick, Gelatt y Vecchi [KIRK83].

Por tanto, el objetivo general de esta tesis es el desarrollo de una estrategia de enfriamiento adaptativo óptimo para el algoritmo del recocido simulado, es decir, una configuración de parámetros de enfriamiento que mantenga un objetivo medio y una desviación típica estadísticamente similares a los del enfriamiento estándar, pero que consiga reducir significativamente el tiempo de ejecución. En el análisis práctico se usan como entornos de prueba (benchmarking) dos problemas clásicos de optimización combinatoria: el problema del viajante comercial y el problema de la asignación cuadrática, evaluando el comportamiento de la estrategia de enfriamiento adaptativo tanto respecto a la eficiencia temporal, como al coste de las soluciones.

Adicionalmente, como ejemplo de aplicación práctica sobre el estudio realizado, se ha elegido el *Problema de Selección de Carteras de Inversión* enmarcado en la Economía, específicamente en la rama de las Finanzas.

1.2. DESCRIPCIÓN GENERAL DE LA ESTRATEGIA DE ENFRIAMIENTO ADAPTATIVO

Para la implementación práctica del algoritmo del recocido simulado debe especificarse un *programa de enfriamiento*, o conjunto de parámetros cuyo objetivo es controlar la evolución del algoritmo. El programa de enfriamiento propuesto por Kirkpatrick, Gelatt y Vecchi [KIRK83] está compuesto por tres parámetros: valor inicial de la temperatura T_0, función de decremento de la temperatura y número de transiciones de estado.

Los dos primeros parámetros especifican la secuencia finita de valores de la temperatura, y es en ellos, en donde se fundamenta la estrategia de enfriamiento adaptativo planteada en esta tesis. El último parámetro indica el número finito de transiciones de estado para cada valor de la temperatura.

El recocido simulado con enfriamiento en su versión estándar [KIRK83] toma como premisa que *el valor inicial de la temperatura debe ser suficientemente elevado como para que se acepte con una cierta probabilidad cercana a 1 cualquier nueva solución generada en una transición de estado*. El planteamiento que se hace en esta investigación es reducir la temperatura inicial T_0 hasta un valor que suponga la ejecución de una fracción porcentual del número medio de ciclos de temperatura efectuados por el algoritmo utilizando la temperatura inicial estándar, sin que ello perjudique la calidad de las soluciones obtenidas.

En la estrategia de enfriamiento diseñada, conjuntamente a la temperatura inicial, interviene la función de decremento de la temperatura. Normalmente, se utiliza una función de decremento exponencial de la forma $T_k = T_0 \cdot \alpha^k$, donde α es una constante un poco menor que la unidad. Por lo general, los valores asignados habitualmente a la constante α están entre 0,8 y 0,99. El segundo objetivo de la investigación es determinar una curva óptima de reducción monótona de la temperatura, y para ello se plantea transformar la constante α de la ecuación exponencial de reducción de la temperatura, $T_k = T_0 \cdot \alpha^k$, en un parámetro que se modifica dinámicamente durante la ejecución, tomando

valores iniciales bajos, α = 0,8, y valores finales cercanos a 1, α = 0,99, de manera que el número medio de ciclos de temperatura efectuados por el algoritmo se mantenga igual que en el caso estándar con parámetro α = 0,95 constante.

En resumen, la característica principal de la estrategia de enfriamiento adaptativo diseñada en la tesis para el algoritmo del recocido simulado es la integración de ambas ideas – la reducción de la temperatura inicial, y el diseño de la curva óptima de reducción monótona de la temperatura en función del cambio dinámico del parámetro α – con el fin de conseguir un programa de enfriamiento que obtenga soluciones finales de calidad equivalente a la del enfriamiento estándar en un tiempo sensiblemente menor.

1.3. PRINCIPALES APORTACIONES

Las principales aportaciones del trabajo realizado a la optimización combinatoria se resumen a continuación:

- *Estudio comparativo riguroso entre programas de enfriamiento del algoritmo meta-heurístico del recocido simulado*. Para ello se evalúa un conjunto representativo de programas de enfriamiento de la literatura: cuatro de enfriamiento monótono multiplicativo, cuatro de enfriamiento monótono aditivo, y uno de enfriamiento no monótono adaptativo [DIAZ08]. Todos ellos se componen al menos de los tres parámetros planteados en el programa de Kirkpatrick, Gelatt y Vecchi: temperatura inicial T_0, función de decremento de la temperatura, y número L de transiciones de estado para cada valor de la temperatura.

- *Desarrollo de una estrategia de enfriamiento adaptativo para el algoritmo del recocido simulado.* Son numerosas las investigaciones que han demostrado las propiedades de convergencia hacia soluciones de alta calidad del recocido simulado aunque, también es conocido el elevado tiempo computacional que presenta este algoritmo. Esta investigación se

ha centrado en reducir el tiempo de ejecución del algoritmo manteniendo el la calidad de las soluciones. Para ello se ha analizado la influencia de la temperatura inicial en el comportamiento del algoritmo usando como referencia el enfriamiento multiplicativo exponencial, y se ha determinado la influencia del parámetro α cuando se modifica dinámicamente en la ecuación de reducción exponencial de la temperatura. Para realizar las pruebas de evaluación, se han utilizado dos problemas clásicos como entornos de prueba o "benchmarking" (el objetivo de las pruebas no es la resolución óptima de estos problemas, sino establecer una comparación rigurosa entre las distintas estrategias de enfriamiento), el problema del viajante sobre un recorrido de 47/80 ciudades y el problema de la asignación cuadrática sobre un total de 47/80 plantas de producción a ubicar sobre 47/80 localizaciones distintas.

- *Desarrollo de una aplicación práctica sobre una importante área de gran interés en la economía: El problema de Optimización de Carteras de Inversión*. La complejidad para resolver este problema de optimización se encuentra en el tipo de limitaciones que contiene y de los activos a incluir en la cartera. Diferentes investigadores han tratado de diseñar modelos más objetivos que reflejen la complejidad de los mercados financieros, como es el problema planteado por los investigadores Crama y Schyns [CRAM01] que utilizaron el recocido simulado para encontrar soluciones al Modelo extendido de Markowitz. Es posible considerar muchas variantes del problema de la selección de carteras de inversión, de diferentes niveles de complejidad. El problema planteado como aplicación práctica en esta investigación es el de construir una cartera que minimice el riesgo para alcanzar una rentabilidad esperada considerando restricciones de cardinalidad, es decir, un problema de optimización combinatoria en el que el espacio de búsqueda aumenta exponencialmente respecto al tamaño del problema, por lo que se hace necesario el uso de algoritmos de optimización aproximada para su resolución. La ejecución del problema se ha realizado sobre 15/50 activos.

1.4. ESQUEMA DE LOS RESTANTES CAPÍTULOS

Esta tesis se ha estructurado en un total de seis capítulos, incluyendo el presente capítulo de introducción. El capítulo dos de la tesis está dedicado a describir el estado actual de las técnicas de resolución de problemas en el área de la optimización combinatoria. El capítulo tres analiza específicamente el recocido simulado debido a que es el algoritmo en el que se centra esta investigación. Asimismo, en este capítulo se realiza un estudio comparativo del comportamiento del recocido simulado respecto a diferentes estrategias de enfriamiento de la literatura. El capítulo cuatro constituye el cuerpo principal de la tesis, en donde se desarrolla la nueva estrategia de enfriamiento adaptativo para el algoritmo del recocido simulado. El capítulo cinco presenta el problema de la selección de carteras de inversión como aplicación práctica al modelo diseñado en la investigación. Finalmente, el capítulo seis incluye las conclusiones sobre el trabajo realizado, así como las líneas futuras de investigación.

El contenido de los restantes cinco capítulos se expone más detalladamente a continuación:

- El capítulo **dos** introduce el área de la optimización combinatoria, comenzando por la definición formal del problema genérico de optimización combinatoria. Se detallan conceptos básicos que se utilizan como base en determinadas técnicas de resolución de problemas y se presentan algunas de las meta-heurísticas más representativas y que han demostrado su eficacia en la resolución de problemas de optimización combinatoria: algoritmos de búsqueda simple, algoritmos trayectoriales poblacionales o de búsqueda múltiple, algoritmos constructivos simples, y algoritmos constructivos poblacionales.

- En el capítulo **tres** se describe detalladamente el algoritmo meta-heurístico del recocido simulado. Se detallan los diferentes programas de enfriamiento utilizados en la comparativa: cuatro de enfriamiento monótono multiplicativo, cuatro de enfriamiento monótono aditivo, y uno de enfriamiento no monótono adaptativo y se expone el análisis de las

pruebas del algoritmo, realizadas sobre dos problemas clásicos de optimización combinatoria: el problema del viajante comercial y el problema de la asignación cuadrática. El análisis práctico de las distintas estrategias de enfriamiento se realiza considerando valores medios y desviaciones típicas de la calidad de las soluciones sobre el marco cualitativo y temporal de referencia establecido por el enfriamiento exponencial de Kirkpatrick, Gelatt y Vecchi. Finalmente, se exponen las conclusiones del estudio realizado.

- El capítulo **cuatro** presenta tres estudios comparativos: el análisis de la influencia de la temperatura inicial en el comportamiento del algoritmo usando como referencia el enfriamiento multiplicativo exponencial, el análisis de la influencia del parámetro α cuando se modifica dinámicamente en la ecuación de reducción exponencial de la temperatura, y un tercer análisis en el que se combina la temperatura inicial y el parámetro α para determinar su influencia en la eficiencia del algoritmo. Estos estudios comparativos consisten en la evaluación estadística de la eficiencia de cada método en la resolución del problema del viajante y el problema de la asignación cuadrática. Como consecuencia de estos estudios, se define la estrategia óptima de enfriamiento adaptativo para la meta-heurística de recocido simulado.

- El capítulo **cinco** describe en detalle el problema de optimización de carteras de inversión seleccionado como aplicación práctica de esta tesis. En primer lugar, se introducen las características generales del modelo de Markowitz y, que hoy en día, suponen la base teórica moderna de la selección de carteras de inversión, y se muestran las particularidades y complejidad del problema. En segundo lugar, se hace un breve repaso de los métodos algorítmicos que han sido utilizados para resolver este tipo de problemas. A continuación, se describe detalladamente el modelo de carteras de inversión a utilizar en la aplicación práctica de la estrategia de enfriamiento adaptativo diseñada en la tesis. Finalmente, se realiza un análisis de las pruebas realizadas con el modelo.

- El capítulo **seis** incluye las conclusiones sobre el trabajo realizado, así como las líneas futuras de investigación que pueden desarrollarse como continuación del mismo.

Capítulo 2: Optimización Combinatoria

"Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber"

Albert Einstein

2.1. OPTIMIZACIÓN COMBINATORIA: CONCEPTOS BÁSICOS

En este primer apartado del capítulo se van a exponer las definiciones, terminología y notación de los conceptos fundamentales de la optimización combinatoria, los cuales constituyen la base necesaria para exponer claramente el estado actual de esta rama de la ciencia. Para este desarrollo conceptual se han tomado como base los trabajos previos de [DÍAZ96] y [DÍAZ11].

2.1.1. Introducción a la optimización combinatoria

Ámbito de la optimización combinatoria

A lo largo de la historia, el ser humano siempre ha buscado la manera más eficiente de realizar sus actividades diarias haciendo un mejor uso de los recursos, con el objetivo de maximizar su rendimiento y/o minimizar los costes asociados, lo cual es especialmente necesario cuando la disponibilidad de tales recursos es limitada. Desde un punto de vista científico, el proceso de búsqueda de la forma más eficiente de realizar una tarea con el fin de

maximizar o minimizar un objetivo o conjunto de objetivos se conoce como *optimización*.

En el ámbito de la ciencia y la tecnología surgen con frecuencia problemas de optimización, es decir, problemas para cuya resolución es necesario realizar un proceso de optimización, tales como los problemas de programación de operaciones de producción, los de planificación de rutas en logística y transporte, o los de diseño y configuración óptima de circuitos electrónicos y redes de telecomunicaciones. La mayoría de estos problemas de optimización son no triviales, es decir, son difíciles de resolver en la práctica, entendiéndose por problema de optimización difícil de resolver aquel para el que no se puede encontrar la solución óptima en un tiempo razonable.

Para poder determinar la mejor solución a un problema de optimización normalmente es conveniente formularlo en términos matemáticos. En general, todo problema de optimización tiene un objetivo (simple o múltiple) representado mediante una función numérica de *n* variables, $f{:}D \subseteq \mathbb{R}^n \to \mathbb{R}$, y un conjunto opcional de restricciones $R = \{r_1,...,r_m\}$, que expresan relaciones de compatibilidad entre las variables y que deben cumplirse para que la solución del problema sea válida. Así, el problema formulado matemáticamente resulta:

Objetivo: $\max/\min\{f(x_1,\ldots,x_n)\}$

Restricciones: $R = \begin{cases} r_1(x_1,...,x_n) = 0 \\ \vdots \\ r_m(x_1,...,x_n) \geq 0 \end{cases}$

Una vez modelizado matemáticamente, es preciso disponer de métodos y técnicas que permitan resolverlo en un tiempo factible. La teoría que proporciona los métodos y técnicas precisos para estudiar estos problemas es la *optimización matemática*. Atendiendo al carácter de las variables que intervienen en el problema, el estudio de la optimización se divide en *optimización continua* (las variables que lo componen son de tipo continuo) y *optimización combinatoria* (las variables toman valores discretos y finitos).

Concretamente, la *optimización combinatoria* tiene por objeto el estudio y la resolución algorítmica de problemas de optimización con restricciones en los que las variables constituyentes son de tipo discreto y finito. Por ello, este tipo de problemas se caracterizan por la existencia de un conjunto finito, aunque muy grande, de soluciones posibles entre las que se debe encontrar aquella que optimice globalmente la función objetivo.

En el resto del apartado se introducen formalmente los conceptos básicos de la optimización combinatoria: problema de optimización combinatoria, soluciones y sus tipos, complejidad algorítmica, tipos notables de problemas, resolución exacta y resolución aproximada mediante heurísticas y meta-heurísticas.

Definición formal del problema genérico de optimización combinatoria

Definición (Problema de optimización combinatoria). Formalmente, un problema de optimización combinatoria es una cuádrupla (X,S,f,R), donde:

- $X = \{x_1,..., x_n\}$, es el conjunto de variables del problema, de dominios discretos y finitos $D_1, \ldots, D_n$. Habitualmente las variables son binarias o toman valores dentro de un subconjunto finito de números naturales, es decir, $D_i = \{0,1\}$ o bien $D_i = \{1,2,..., p\}$.
- $S = D_1 \times \cdots \times D_n$, es el espacio de soluciones del problema, formado por todas las asignaciones posibles de valores a las variables dentro de sus respectivos dominios. Así, una solución arbitraria se representa mediante un vector $\boldsymbol{x} = (x_1,\ldots,x_n) \in S$.
- $f : S \to \mathbb{R}$, es la función objetivo a optimizar, que asigna un número real a cada posible solución del problema, $\boldsymbol{x} = (x_1,\ldots,x_n) \in S \to y = f(x_1,\ldots,x_n) \in \mathbb{R}$.
- $R = \{r_1,... ,r_m\}$, es el conjunto de restricciones, cada una de las cuales determina una relación de compatibilidad entre las variables del problema. Generalmente, las restricciones se expresan mediante

igualdades o desigualdades numéricas, como en la programación matemática clásica: $r_i(x_1,\ldots,x_n)=0$, $r_j(x_1,\ldots,x_n)\geq 0$.

El objetivo del problema consiste en encontrar una asignación simultánea de valores de las variables, $\boldsymbol{s}_{\text{opt}}=(s_1,\ldots,s_n)\in S$, que optimice (minimice o maximice) globalmente la función objetivo *f*, de tal forma que se satisfagan todas las restricciones impuestas entre las variables. Tal solución $\boldsymbol{s}_{\text{opt}}$ se denomina solución óptima global, siendo $f(\boldsymbol{s}_{\text{opt}})$ el objetivo óptimo a conseguir. Asimismo, el conjunto de soluciones que verifican todas las restricciones *R* del problema, se denomina espacio de soluciones viables, y se denota $S^*\subseteq S$.

2.1.2. Soluciones de un problema de optimización combinatoria

En la definición del problema genérico de optimización combinatoria como una cuádrupla (*X*,*S*,*f*,*R*), se ha utilizado el término *solución* del problema para referirse a toda asignación de valores a las variables dentro de sus respectivos dominios, y el término *solución óptima global* para referirse a la solución que verifica el conjunto de restricciones *R* y optimiza globalmente la función objetivo *f*. Además de estas dos acepciones, es posible considerar otros tipos de soluciones de un problema de optimización combinatoria cuya terminología es habitual en la descripción de las técnicas de resolución. A continuación se definen formalmente los diferentes tipos de soluciones.

Soluciones candidatas

Definición (Solución candidata). Dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), se denomina *solución candidata* a una asignación concreta de valores a todas las variables del problema, $\boldsymbol{s}=(x_1=s_1,\ldots,x_n=s_n)\in S$. Si la solución candidata verifica el conjunto de restricciones *R* se llama *solución candidata viable*, y en caso contrario *solución candidata inviable*.

Observación. En lo sucesivo, el término *solución candidata* se interpretará por defecto como *solución candidata viable*.

Soluciones candidatas vecinas

Definición (Solución candidata vecina). Sean un problema de optimización combinatoria (*X*,*S*,*f*,*R*), una solución candidata $\boldsymbol{s} = (s_1,...,s_n) \in S$, y una función de distancia $d : S \times S \to \mathbb{R}$. Se denomina *solución candidata vecina o próxima* a $\boldsymbol{s}$ respecto a una distancia $\delta > 0$ a toda solución candidata $\boldsymbol{x} = (x_1,...,x_n) \in S$ que verifique $d(\boldsymbol{x},\boldsymbol{s}) \leq \delta$. Si la solución candidata vecina verifica el conjunto de restricciones *R* se llama *solución candidata vecina viable*, y en caso contrario *solución candidata vecina inviable*.

Observación. En lo sucesivo, el término *solución candidata vecina* se interpretará por defecto como *solución candidata vecina viable*.

En la definición de las soluciones candidatas vecinas es especialmente relevante la función de distancia considerada $d : S \times S \to \mathbb{R}$, la cual obviamente dependerá del problema a resolver. Los casos habituales de distancias entre soluciones son la distancia euclídea, y la distancia de Hamming generalizada.

Definición (Distancia euclídea). Dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), se define la *distancia euclídea* entre dos soluciones candidatas $\boldsymbol{x} = (x_1,...,x_n), \boldsymbol{y} = (y_1,...,y_n) \in S$ como el número real dado por:

$$d(\boldsymbol{x},\boldsymbol{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Por ejemplo, la distancia euclídea entre los vectores de enteros positivos $\boldsymbol{x} = (2,1,2,1), \boldsymbol{y} = (1,1,1,5) \in \mathbb{N}^4$ es:

$$d(\boldsymbol{x},\boldsymbol{y}) = \sqrt{(2-1)^2 + (1-1)^2 + (2-1)^2 + (1-5)^2} = \sqrt{18}$$

Definición (Distancia de Hamming generalizada). Dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), se define la *distancia de Hamming*

generalizada entre dos soluciones candidatas $\boldsymbol{x} = (x_1,..., x_n), \boldsymbol{y} = (y_1,..., y_n) \in S$ como el número natural dado por:

$$d(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{n} \varepsilon(x_i, y_i)$$

donde ε es un valor binario que indica la existencia de error o diferencia en el valor de las variables argumento:

$$\varepsilon(x_i, y_i) = \begin{cases} 0 & \text{si } x_i = y_i \\ 1 & \text{si } x_i \neq y_i \end{cases}$$

Por ejemplo, la distancia de Hamming entre los vectores de enteros positivos $\boldsymbol{x} = (2,1,5,1), \boldsymbol{y} = (1,1,1,1) \in \mathbb{N}^4$ es:

$$d(\boldsymbol{x},\boldsymbol{y}) = 1 + 0 + 1 + 0 = 2$$

La distancia euclídea se utiliza en problemas en los que las soluciones pueden interpretarse como puntos de un espacio euclídeo, lo que es más habitual en la optimización continua. En este caso, también pueden utilizarse otras distancias métricas como la distancia Manhattan (suma de las diferencias en valor absoluto entre coordenadas correspondientes) o la distancia del máximo (máximo de las diferencias en valor absoluto entre coordenadas correspondientes). Sin embargo, esto no es muy habitual en problemas de optimización combinatoria, en los que las variables son enteras y representan objetos o lugares, estableciéndose una asociación arbitraria entre el objeto representado y el número natural que lo representa, por lo que la distancia euclídea y sus variantes métricas pueden resultar inapropiadas. En estos casos las medidas de distancia más convenientes son las que se basan en el número de diferencias entre ambas soluciones, tal como la distancia de Hamming.

Definición (Vecindad). Dados un problema de optimización combinatoria (X,S,f,R), una solución candidata $\boldsymbol{s} = (s_1,..., s_n) \in S$, y una función de distancia $d : S \times S \to \mathbb{R}$, el conjunto de soluciones candidatas vecinas a una solución $\boldsymbol{s} = (s_1,..., s_n) \in S$ respecto a una distancia $\delta > 0$, denotado $S(\boldsymbol{s},\delta)$, se llama *vecindad* o *proximidad* de $\boldsymbol{s}$:

$$S(\boldsymbol{s},\delta) = \{\boldsymbol{x} = (x_1,...,x_n) \in S \mid d(\boldsymbol{x},\boldsymbol{s}) \leq \delta\}$$

Asimismo, el conjunto de soluciones candidatas vecinas a ***s*** que verifican todas las restricciones *R* del problema, se denomina *vecindad viable*, y se denota $S^*(\boldsymbol{s},\delta)$, verificándose que $S^*(\boldsymbol{s},\delta) \subseteq S(\boldsymbol{s},\delta)$.

Debido a su importancia práctica en la optimización combinatoria, la vecindad de soluciones candidatas basada en la distancia de Hamming recibe el nombre de k-proximidad.

Definición (k-proximidad). Dados un problema de optimización combinatoria (*X*,*S*,*f*,*R*), una solución candidata $\boldsymbol{s} = (s_1,...,s_n) \in S$, y la función de distancia de Hamming generalizada, el conjunto de soluciones candidatas vecinas a una solución $\boldsymbol{s} = (s_1,...,s_n) \in S$ respecto a una distancia *k*, es decir, el conjunto de soluciones candidatas que difieren de ***s*** como máximo en el valor de *k* variables, se llama *k-proximidad* de ***s***.

Las definiciones anteriores de vecindad de una solución candidata se basan en el concepto de función de distancia, ya sea la distancia euclídea, la distancia de Hamming (k-proximidad), o cualquier otra distancia métrica que pudiera establecerse. Sin embargo, una forma adicional de considerar la vecindad de una solución candidata, muy habitual en los problemas de optimización combinatoria, es la vecindad respecto a una función transformativa simple. Las funciones transformativas simples convierten una solución candidata en otra solución vecina, también denominada solución sucesora, alterando los valores de los variables de una forma simple y aleatoria, pero adaptada al problema concreto de optimización, es decir, a los dominios de las variables y a las restricciones del problema.

Definición (Vecindad respecto a una función transformativa simple). Dados un problema de optimización combinatoria (*X*,*S*,*f*,*R*), una solución candidata $\boldsymbol{s} = (s_1,...,s_n) \in S$, y una función transformativa simple $g : S \to S$, el conjunto de soluciones candidatas que pueden obtenerse a partir de ***s*** aplicando la función de transformación simple *g*, denotado *S*(***s***,*g*), se llama *vecindad respecto a la función transformativa simple g* de ***s***:

$$S(\boldsymbol{s},g) = \{\boldsymbol{x} = (x_1,..., x_n) \in S \mid \boldsymbol{x} = g(\boldsymbol{s})\}$$

Por ejemplo, si se considera el vector de enteros positivos $\boldsymbol{x} = (2,1,2,1) \in \mathbb{N}^4$ y se toma como función de transformación simple *g* el intercambio de valores de dos variables tomadas al azar, se observa que el vector $\boldsymbol{y} = (2,1,1,2) \in \mathbb{N}^4$ pertenece a la vecindad de ***x*** respecto a *g*, ya que basta un intercambio de valores entre las variables x_3 y x_4 para que el vector ***x*** se convierta en el vector ***y***.

Las soluciones candidatas y su vecindad se utilizan más propiamente en los métodos de resolución aproximada de tipo transformativo, explicados más adelante en el apartado 2.1.6 de técnicas de resolución.

Soluciones parciales

Definición (Solución parcial). Dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), se denomina *solución parcial* a una asignación concreta de valores a *k* variables del problema, $\boldsymbol{s} = (x_1 = s_1,..., x_k = s_k, x_{k+1},..., x_n) \in S$, siendo $k < n$, quedando libres (sin asignar) las restantes $n - k$ variables. Si la solución parcial verifica el subconjunto de restricciones que afectan sólo a las *k* variables asignadas se llama *solución parcial viable*, y en caso contrario *solución parcial inviable*.

Observación. En lo sucesivo, el término *solución parcial* se interpretará por defecto como *solución parcial viable*.

Las soluciones parciales se utilizan más propiamente en los métodos de resolución aproximada de tipo constructivo, explicados más adelante en el apartado 2.1.6 de técnicas de resolución.

Soluciones óptimas

Definición (Solución óptima global). Dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), se dice que una solución candidata viable $\boldsymbol{s} = (s_1,...,s_n) \in S^*$ es un *óptimo global* de *f* si y sólo si para toda solución candidata viable $\boldsymbol{x} = (x_1,...,x_n) \in S^*$ se cumple que $f(\boldsymbol{s}) \leq f(\boldsymbol{x})$ en caso de minimización, o $f(\boldsymbol{s}) \geq f(\boldsymbol{x})$ en caso de maximización.

Definición (Solución óptima local). Dados un problema de optimización combinatoria (*X*,*S*,*f*,*R*), y una función de distancia $d : S \times S \rightarrow \mathbb{R}$, se dice que una solución candidata viable $\boldsymbol{s} = (s_1,...,s_n) \in S^*$ es un *óptimo local* de *f* con respecto a la vecindad $S^*(\boldsymbol{s},\delta) = \left\{ \boldsymbol{x} = (x_1,...,x_n) \in S^* \mid d(\boldsymbol{x},\boldsymbol{s}) \leq \delta \right\}$ si y sólo si para toda solución candidata vecina viable $\boldsymbol{x} = (x_1,...,x_n) \in S^*(\boldsymbol{s},\delta)$ se cumple que $f(\boldsymbol{s}) \leq f(\boldsymbol{x})$ en caso de minimización, o $f(\boldsymbol{s}) \geq f(\boldsymbol{x})$ en caso de maximización.

Observación. La definición de solución óptima local es igualmente válida para vecindades definidas respecto a una función de transformación simple $g : S \rightarrow S$.

2.1.3. Complejidad algorítmica

Órdenes de complejidad de un algoritmo

El análisis de algoritmos es una parte fundamental de la algoritmia matemática cuyo objetivo es determinar el coste computacional de la ejecución de cada algoritmo, en los casos medio y peor, en función del tamaño del problema que el algoritmo resuelve (por ejemplo, en función del número *n* de datos de entrada del algoritmo).

En general, el coste computacional de un algoritmo se determina mediante funciones de variable discreta *n* (sucesiones) que miden diversas magnitudes

como la cantidad de memoria o el tiempo de ejecución utilizados, de acuerdo con el tamaño del problema, representado por dicha variable *n*. Precisamente, el análisis temporal es especialmente interesante cuando el algoritmo se aplica a problemas de gran tamaño. Formalmente, el análisis temporal consiste en estudiar el comportamiento temporal asintótico de un algoritmo, cuando el tamaño *n* del problema tiende a infinito.

Del análisis matemático real se sabe que un conjunto de sucesiones que comparten un mismo comportamiento asintótico tienen el mismo *orden de complejidad*, existiendo cinco órdenes fundamentales de complejidad de las sucesiones infinitas u órdenes fundamentales del infinito:

- Orden de las sucesiones logarítmicas: $\underset{n\to+\infty}{\text{ord}}\left(\log_a(n)\right)$, con $a>1$.
- Orden de las sucesiones potenciales: $\underset{n\to+\infty}{\text{ord}}\left(n^b\right)$, con $b>0$.
- Orden de las sucesiones exponenciales: $\underset{n\to+\infty}{\text{ord}}\left(c^n\right)$, con $c>1$.
- Orden de las sucesiones factoriales: $\underset{n\to+\infty}{\text{ord}}\left(n!\right)$.
- Orden de las sucesiones potenciales-exponenciales: $\underset{n\to+\infty}{\text{ord}}\left(n^{dn}\right)$, con $d>0$.

Los órdenes fundamentales establecen categorías en la velocidad de crecimiento hacia el infinito de las sucesiones infinitas (comportamiento asintótico), estableciéndose la siguiente relación entre los órdenes fundamentales:

$$\underset{n\to+\infty}{\text{ord}}\left(\log_a(n)\right) << \underset{n\to+\infty}{\text{ord}}\left(n^b\right) << \underset{n\to+\infty}{\text{ord}}\left(c^n\right) << \begin{cases} \underset{n\to+\infty}{\text{ord}}\left(n!\right) << \underset{n\to+\infty}{\text{ord}}\left(n^{dn}\right) & d \geq 1 \\ \underset{n\to+\infty}{\text{ord}}\left(n^{dn}\right) << \underset{n\to+\infty}{\text{ord}}\left(n!\right) & 0 < d < 1 \end{cases}$$

Así, las sucesiones que más lentamente crecen hacia el infinito son las logarítmicas, mientras que las sucesiones que más rápidamente crecen hacia el infinito son las factoriales y las potenciales-exponenciales.

Respecto al análisis de algoritmos, los de complejidad temporal potencial (polinómica) o inferior pueden utilizarse en la práctica para resolver problemas de tamaño grande o muy grande, mientras que los algoritmos de complejidad temporal exponencial o superior sólo pueden aplicarse en la práctica para resolución de problemas de tamaño pequeño, ya que el crecimiento del tiempo de ejecución es explosivo. Como ejemplo ilustrativo se ofrece la tabla de valores de la sucesión exponencial $a_n = 2^n$:

n	**2^n**
1	2
2	4
3	8
10	1024
20	1048576
30	1073741824
50	1125899906842624
100	1267650600228229401496703205376

Tabla 2.1: Crecimiento de la sucesión exponencial $a_n = 2^n$

Obsérvese que para un valor aún bastante reducido como $n = 50$, y suponiendo como unidad de tiempo el milisegundo, el tiempo de ejecución de un algoritmo de complejidad exponencial de base 2, $O(2^n)$ en notación de Landau, es superior a mil billones de milisegundos, es decir, más de treinta y cinco mil años de cálculo.

Clasificación de los problemas según su complejidad algorítmica

Los problemas de optimización combinatoria se clasifican de acuerdo con el coste computacional (recursos necesarios) del mejor algoritmo que resuelve el problema. Como se ha mencionado, aunque se utilizan diversas magnitudes para medir el coste computacional de un algoritmo, la más usual es el tiempo de ejecución. Por ello, lo que se plantea específicamente en el análisis y

clasificación de los problemas de optimización combinatoria es determinar el orden de complejidad temporal de:

a. El mejor algoritmo conocido que verifica si una solución concreta del problema es la solución óptima global (algoritmo de verificación).

b. El mejor algoritmo conocido que obtiene la solución óptima global del problema (algoritmo de resolución).

De acuerdo con lo anterior, los problemas de optimización combinatoria se dividen en las siguientes clases según su complejidad temporal, es decir, según el tiempo computacional que emplea el mejor algoritmo conocido para su verificación y/o resolución en función del tamaño del problema:

- Problemas de complejidad temporal potencial determinista, o problemas P (*Deterministic Polynomial Time Problems*): Clase de problemas para los que existe un algoritmo de resolución de complejidad temporal potencial o inferior, es decir, un algoritmo que en un tiempo potencial encuentra la solución óptima global del problema.

- Problemas de complejidad temporal potencial no determinista, o problemas NP (*Nondeterministic Polynomial Time Problems*): Clase de problemas para los que existe un algoritmo de verificación de complejidad temporal potencial o inferior, es decir, un algoritmo que en un tiempo potencial verifica si una solución arbitraria es la solución óptima global del problema. Se verifica que la clase P está contenida en la clase NP, ya que si existe un algoritmo resolutivo del problema de complejidad temporal potencial (clase P), puede ejecutarse ese mismo algoritmo para obtener la solución óptima y compararla con la solución a verificar, es decir, el propio algoritmo resolutivo es un algoritmo de verificación de complejidad temporal potencial (clase NP).

- Problemas completos de complejidad temporal potencial no determinista, o problemas NP-completos (*Nondeterministic Polynomial Time Complete Problems*): Clase de problemas para los que sí se conoce un algoritmo de verificación de complejidad temporal potencial

(son por ello una subclase de la clase NP), pero para los que no se ha podido obtener aún un algoritmo de resolución de complejidad temporal potencial, con la conjetura de que estos problemas sólo pueden resolverse mediante algoritmos de complejidad superior a la potencial. Esto quiere decir que no se ha podido demostrar formalmente que no exista un algoritmo resolutivo de complejidad potencial, pero se conjetura que realmente no existe. Durante los años setenta, el desarrollo de la teoría de la computación ha proporcionado una formulación rigurosa a esta conjetura, resultando la llamada teoría de la NP-completud [GARE79]. En la práctica, la resolución exacta de este tipo de problemas es imposible en tiempos de ejecución razonables.

- Problemas duros de complejidad temporal potencial no determinista, o problemas NP-duros (*Nondeterministic Polynomial Time Hard Problems*): Clase de problemas para los que no se conoce siquiera un algoritmo de verificación de complejidad temporal potencial o inferior. Son por ello problemas independientes de los anteriores grupos (no forman una subclase de la clase NP), y se supone que son tan difíciles o incluso más que los problemas NP-completos.

En resumen, los problemas P poseen algoritmos de resolución de complejidad temporal potencial, los problemas NP poseen algoritmos de verificación de complejidad temporal potencial, los problemas NP-completos poseen algoritmos de verificación de complejidad temporal potencial, pero se conjetura que sólo pueden resolverse mediante algoritmos de complejidad temporal superior a la potencial, y finalmente los problemas NP-duros no poseen algoritmos conocidos de verificación (ni por supuesto de resolución) de complejidad temporal potencial o inferior. La relación entre estas clases de problemas puede expresarse con el siguiente diagrama de Venn:

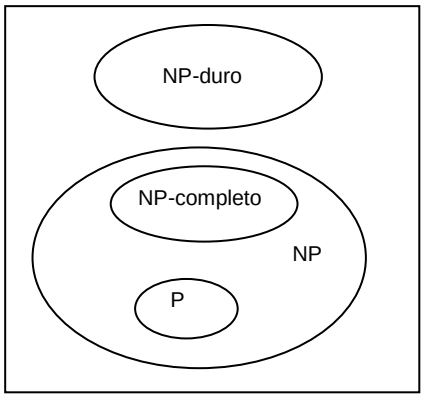

Figura 2.1: Relación entre las clases P, NP, NP-completo, NP-duro

Con respecto a los problemas de optimización combinatoria, la gran mayoría son problemas NP-completos o NP-duros, es decir, no se conoce ningún algoritmo de resolución de complejidad temporal potencial o inferior y se conjetura que tal algoritmo no existe. En estos casos, para obtener la solución óptima global del problema es necesario considerar algoritmos de complejidad superior a la potencial, generalmente basados en la enumeración sistemática de soluciones, imposibles de usar en la práctica cuando el tamaño del problema no es reducido. Por otra parte, un problema de optimización combinatoria de esta clase (sin algoritmo de resolución de complejidad potencial o inferior) es NP-completo si se conoce el valor óptimo global de la función objetivo (y obviamente ésta se evalúa mediante un algoritmo de complejidad potencial o inferior), ya que entonces resulta trivial verificar una solución: basta evaluar la función objetivo para esa solución y comprobar si el resultado es igual al óptimo global. Por el contrario si no se conoce el valor óptimo global de la función objetivo y la verificación sólo es posible a través del propio algoritmo exacto de resolución, cosa mucho más habitual, el problema será NP-duro.

2.1.4. Tipos notables de problemas de optimización combinatoria

Los problemas más importantes de optimización combinatoria se originan en diversas áreas de la ingeniería y sobre todo en el ámbito de la organización industrial y logística. Pueden clasificarse en los siguientes tipos:

- Problemas de planificación de rutas, en los que se trata de encontrar una ruta que visite un conjunto de lugares predeterminados con el objetivo de minimizar la distancia recorrida o el tiempo empleado, tales como el *problema del viajante comercial*, o el *problema de la planificación de rutas de distribución*.

- Problemas de configuración, en los que se trata de disponer una serie de objetos en una determinada forma o estructura con el objetivo de optimizar el espacio o los recursos utilizados, tales como el *problema del empaquetado*, el *problema de la partición*, o el *problema de la asignación cuadrática*.

- Problemas de selección, en los que se deben elegir varios objetos de un conjunto de modo que se optimice algún criterio de selección o relevancia, tales como el *problema de la mochila*, o el *problema de la máxima diversidad*.

- Problemas de planificación de tareas, en los que se debe asignar a una serie de tareas o actividades los recursos (personas, máquinas) necesarios para su realización óptima respecto al tiempo o al coste, tales como el *problema de la programación de operaciones de producción*.

Los problemas mencionados en las cuatro categorías anteriores han servido como referencia en sus versiones más simplificadas o académicas para el estudio de nuevas técnicas de optimización combinatoria durante las últimas décadas, por lo que se consideran problemas fundamentales de la optimización combinatoria. Además, sus múltiples generalizaciones y variantes constituyen la gran mayoría de los problemas reales de optimización que aparecen en las

aplicaciones prácticas. Por ello, se realiza a continuación una descripción formal de cada uno de estos problemas fundamentales.

1. *Problema del viajante comercial*.

Definición (Problema del viajante comercial). El problema del viajante comercial (*Travelling Salesman Problem*, TSP) consiste en encontrar la ruta más corta de recorrido cíclico de *n* ciudades de forma que cada ciudad se visite una sola vez. En general, el problema puede modelizarse mediante un grafo etiquetado completo no dirigido $K_n = (V, A, p, w)$ de $\#V = n$ vértices y $\#A = \frac{1}{2}n(n-1)$ aristas, en el que los vértices representan las ciudades a visitar, las aristas representan las carreteras entre las ciudades, y los pesos de las aristas las distancias entre cada par de ciudades. De acuerdo con esta modelización, la ruta óptima a determinar viene dada por el ciclo hamiltoniano de longitud mínima del grafo K_n. Desde el punto de vista de la optimización combinatoria, el problema del viajante comercial es formalmente una cuádrupla (*X*,*S*,*f*,*R*), donde:

- $X = \{x_1, ..., x_n\}$, es el conjunto de variables del problema, en el que cada variable x_i representa la posición *i* de la ruta a recorrer. El dominio finito común para las *n* variables es $D = \{1, 2, ..., n\}$, que representa al conjunto de las *n* ciudades a visitar.

- $S = D^n$, es el espacio de soluciones del problema, de tamaño $\#S = \#D^n = n^n$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1, \ldots, x_n) \in S$.

- $f : S \to \mathbb{R}$, es la función objetivo a minimizar, o distancia de recorrido cíclico de las n ciudades, determinada por:

$$f(\boldsymbol{x}) = f(x_1, \ldots, x_n) = \sum_{i=1}^{n} d\left(x_i, x_{(i \bmod n)+1}\right)$$

  donde $d(x_i, x_j)$ es la distancia existente entre las ciudades x_i y x_j.

- $R = \{r_1, \ldots, r_n\}$, es el conjunto de restricciones:

$$r_1 : x_1 = 1$$

$$r_i : \forall j \in \{1,..., i-1\} : x_i \neq x_j \quad (i = 2,..., n)$$

La circularidad de la ruta buscada permite fijar la primera variable x_1 con la ciudad 1 (restricción r_1), y asignar libremente las restantes ciudades 2 a n a las variables x_2 a x_n, teniendo en cuenta que no puede repetirse ninguna ciudad en la ruta (restricciones r_2 a r_n). Por tanto, las soluciones viables son permutaciones de las n ciudades con la ciudad 1 como primera ciudad a visitar. Así, dada una ruta $\boldsymbol{x}$ que verifique las restricciones, cualquier otra ruta cíclica que pase por las n ciudades en igual orden (y por tanto con igual valor objetivo asociado) será una rotación de $\boldsymbol{x}$. Nota: La rutas correspondientes en orden inverso se consideran aquí diferentes, aunque el valor objetivo asociado sea el mismo.

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1,..., s_n) \in S$, que minimice globalmente la función objetivo f, de tal forma que se satisfagan todas las restricciones impuestas entre las variables. El espacio de soluciones viables es el conjunto de las permutaciones circulares de n ciudades: $S^* = \left\{\boldsymbol{x} = (1, x_2,..., x_n) \in S \mid \forall i \in \{2,..., n\}, \forall j \in \{1,..., i-1\} : x_i \neq x_j\right\}$, de tamaño $\#S^* = (n-1)!$.

El problema del viajante comercial es un problema NP-duro [KARP72], y ha sido extensamente estudiado en las últimas décadas. Existen diversas variantes del problema, tales como: TSP euclídeo, en el que la distancia entre las ciudades de la ruta es la distancia euclídea en el plano; TSP rectilíneo, en el que la distancia entre las ciudades de la ruta es la distancia Manhattan en el plano; TSP máximo, en el que la distancia entre las ciudades de la ruta es la métrica del máximo en el plano; TSP no euclídeo, en el que la distancia entre las ciudades de la ruta no cumple la propiedad de desigualdad triangular, es decir, cuando el trayecto directo entre dos ciudades puede ser más largo que un trayecto indirecto a través de una ciudad intermedia, lo que puede ocurrir por ejemplo cuando la distancia utilizada se mide en tiempo en vez de en

unidades lineales; TSP asimétrico, en el que la distancia entre las ciudades de la ruta no cumple la propiedad de simetría, es decir, cuando el trayecto directo entre dos ciudades en un sentido puede ser más largo que el trayecto en sentido contrario, lo que puede ocurrir por ejemplo cuando en el trayecto se recorren caminos de un solo sentido; TSP múltiple, en el que las *n* ciudades son visitadas por *m* viajantes, con $m < n$.

2. *Problema de la planificación de rutas de distribución*.

Definición (Problema de la planificación de rutas de distribución). El problema de la planificación de rutas de distribución (*Vehicle Routing Problem*, VRP) consiste en encontrar el conjunto de rutas cíclicas de reparto para *m* vehículos iguales de capacidad *Q* con inicio y fin en un almacén o depósito común, que supongan el coste mínimo de abastecimiento de *n* clientes con sendos pedidos de cantidades p_i, de forma que cada cliente se visite una sola vez. Obviamente, en este problema se debe tomar como hipótesis que la capacidad total de la flota de vehículos es igual o superior a la demanda total de los clientes, $\sum_{i=1}^{n} p_i \leq mQ$, ya que de lo contrario el reparto no sería posible. En general, el problema puede modelizarse mediante un grafo etiquetado completo no dirigido $K_{n+1} = (V, A, p, w)$ de $\#V = n+1$ vértices y $\#A = \frac{1}{2}n(n+1)$ aristas, en el que el primer vértice es el almacén y los restantes vértices representan los clientes a visitar, las aristas representan las carreteras entre los lugares donde están el almacén y los clientes, y los pesos de las aristas las distancias entre cada par de lugares. Por tanto, el VRP puede verse como una generalización del TSP con múltiples rutas con la particularidad de que las rutas convergen en un punto común (almacén). Desde el punto de vista de la optimización combinatoria, el problema de la planificación de rutas de vehículos es formalmente una cuádrupla (*X*,*S*,*f*,*R*), donde:

- $X = \{x_1, ..., x_{n+m}\}$, es el conjunto de variables del problema, en el que cada variable x_i, con $i \in \{1, ..., n\}$, representa la posición *i* de la secuencia general de clientes a visitar, y cada variable x_{n+j}, con $j \in \{1, ..., m\}$, representa el número de clientes consecutivos de dicha secuencia que

debe visitar el vehículo *j* (contados a partir del siguiente cliente al último visitado por el vehículo *j*–1). El dominio finito común para las variables x_1 a x_n es el conjunto de los *n* clientes a visitar $D_1 = \{1,2,...,n\}$, y el dominio finito común para las variables x_{n+1} a x_{n+m} es el conjunto $D_2 = \{0,1,...,n\}$.

- $S = D_1^n \times D_2^m$, es el espacio de soluciones del problema, de tamaño $\#S = \#\left(D_1^n \times D_2^m\right) = n^n (n+1)^m$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1,\ldots,x_{n+m}) \in S$.
- $f : S \to \mathbb{R}$, es la función objetivo a minimizar, o distancia total recorrida por los *m* vehículos para visitar los *n* clientes, determinada por:

$$f(\boldsymbol{x}) = f(x_1,\ldots,x_{n+m}) = \sum_{j=1}^{m}\left(d\left(0, x_{k_j}\right) + \sum_{i=k_j}^{k_j + x_{n+j} - 2} d(x_i, x_{i+1}) + d\left(x_{k_j + x_{n+j} - 1}, 0\right)\right)$$

donde $d(x_i, x_j)$ es la distancia existente entre los lugares x_i y x_j, 0 representa al almacén, y la variable k_j indica la posición de comienzo de la ruta asociada al vehículo *j* en la secuencia general de clientes, la cual se define:

$$k_j = \begin{cases} 1 & \text{si } j = 1 \\ 1 + \sum_{k=1}^{j-1} x_{n+k} & \text{si } j > 1 \end{cases}$$

- $R = \{r_1,\ldots,r_{n+m}\}$, es el conjunto de restricciones:

$$r_1 : \sum_{j=1}^{m} x_{n+j} = n$$

$$r_i : \forall j \in \{1,\ldots,i-1\} : x_i \neq x_j \quad (i = 2,\ldots,n)$$

$$r_{n+j} : \sum_{i=k_j}^{k_j + x_{n+j} - 1} p_{x_i} \leq Q \quad (j = 1,\ldots,m)$$

De acuerdo con el significado y el dominio de las variables x_{n+1} a x_{n+m}, la restricción r_1 asegura que cada pedido de cliente es servido por un único vehículo de la flota. Asimismo, las restricciones r_2 a r_n que relacionan las

variables x_1 a x_n aseguran que cada cliente se visita una sola vez. Finalmente, las restricciones r_{n+1} a r_{n+m}, establecen que cada vehículo transporta una cantidad de pedidos no superior a su capacidad, donde k_j indica la posición de comienzo de la ruta asociada al vehículo *j* en la secuencia general de clientes.

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1,...,s_{n+m}) \in S$, que minimice globalmente la función objetivo *f*, de tal forma que se satisfagan todas las restricciones impuestas entre las variables. El espacio de soluciones viables es el conjunto $S^* = \{\boldsymbol{x} = (x_1,...,x_{n+m})\} \subseteq S$, formado por las permutaciones de los *n* clientes para las variables x_1 a x_n, y las variaciones con repetición con *m* elementos que suman *n* del conjunto $D_2 = \{0,1,...,n\}$ para las variables x_{n+1} a x_{n+m}, siendo el tamaño de dicho espacio de soluciones viables $\#S^* = n!\left(\frac{(m+n-1)!}{n!(m-1)!}\right)$. Así, dada una solución viable $\boldsymbol{x} = (x_1,\ldots,x_{n+m}) \in S^*$ se pueden obtener las *m* rutas cíclicas de reparto de la siguiente forma: Si para un $j \in \{1,...,m\}$ se cumple que $x_{n+j} \neq 0$ y k_j es la posición de comienzo de la ruta asociada al vehículo *j* en la secuencia general de clientes, la ruta correspondiente es $\boldsymbol{r}_j = (0, x_{k_j},\ldots,x_{k_j+x_{n+j}-1})$.

- Como el TSP, y precisamente por ser una generalización de éste, el problema de la planificación de rutas de distribución es un problema NP-duro. Ha sido muy estudiado en las últimas décadas, debido especialmente a su frecuente aparición como problema real en todo tipo de empresas logísticas y de distribución. De hecho, existen múltiples variantes o generalizaciones del problema, las cuales pueden combinarse entre sí, tales como: VRP con vehículos homogéneos, en el que los *m* vehículos de reparto son iguales, por lo que todos poseen una capacidad común *Q* (caso básico); VRP con vehículos heterogéneos, en el que los *m* vehículos de reparto son diferentes, de manera que cada uno posee una capacidad específica q_j; VRP con múltiples almacenes, en el que los *n* clientes son abastecidos por varios almacenes cada uno

con su correspondiente flota de vehículos; VRP con ventanas temporales, en el que cada cliente tiene asociado un intervalo temporal de servicio fuera del cual no puede (o le resulta inconveniente) recibir el pedido; VRP con entregas y retornos, en el que los pedidos de clientes no son únicamente de entrega de mercancía sino también de recogida o retorno, por ejemplo devoluciones de mercancía defectuosa o retornos de envases reutilizables.

3. *Problema del empaquetado*.

Definición (Problema del empaquetado). El problema del empaquetado (*Bin Packing Problem*, BPP) consiste en determinar el número mínimo de cajas o contenedores iguales de capacidad *V* en los que se pueden empaquetar *n* objetos de tamaños diferentes v_i. Obviamente, en este problema se debe tomar como hipótesis que la capacidad individual de los contenedores es igual o superior al tamaño de cada objeto, $\forall i \in \{1,...,n\} : v_i \leq V$, ya que de lo contrario el empaquetado no sería posible. Desde el punto de vista de la optimización combinatoria, el problema del empaquetado es una cuádrupla (*X*,*S*,*f*,*R*), donde:

- $X = \{x_1,...,x_n\}$, es el conjunto de variables del problema, en el que cada variable x_i representa la posición *i* de la secuencia general de empaquetado de los objetos. El dominio finito común para las *n* variables es el conjunto de los *n* objetos a empaquetar $D = \{1,2,...,n\}$.

- $S = D^n$, es el espacio de soluciones del problema, de tamaño $\#S = \#D^n = n^n$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1,\ldots,x_n) \in S$.

- $f : S \to \mathbb{R}$, es la función objetivo a minimizar, o número de contenedores utilizados para empaquetar los *n* objetos, determinada por:

$$f(\boldsymbol{x}) = f(x_1,\ldots,x_n) = \sum_{i=1}^{n} y_i$$

donde cada y_i es una variable binaria que indica si en la posición *i* de la secuencia general de empaquetado de los objetos comienza a llenarse un nuevo contenedor ($y_i = 1$), o no ($y_i = 0$), y se define:

$$y_i = \begin{cases} 1 & \text{si } i = 1 \\ 0 & \text{si } i > 1 \wedge \sum_{j=\max\{k \in \{1,\ldots,i-1\} | y_k = 1\}}^{i} v_{x_j} \leq V \\ 1 & \text{si } i > 1 \wedge \sum_{j=\max\{k \in \{1,\ldots,i-1\} | y_k = 1\}}^{i} v_{x_j} > V \end{cases}$$

- $R = \{r_1, \ldots, r_{n-1}\}$, es el conjunto de restricciones:

$$r_i : \forall j \in \{1,\ldots,i\} : x_{i+1} \neq x_j \quad (i = 1,\ldots,n-1)$$

Las restricciones r_1 a r_{n-1} que relacionan las variables x_1 a x_n aseguran que cada objeto se empaqueta una sola vez.

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1,\ldots,s_n) \in S$, que minimice globalmente la función objetivo *f*, de tal forma que se satisfagan todas las restricciones impuestas entre las variables. El espacio de soluciones viables es el conjunto $S^* = \{\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in S \mid \forall i \in \{1,\ldots,n-1\}, \forall j \in \{1,\ldots,i\} : x_{i+1} \neq x_j\}$, formado por las permutaciones de los *n* objetos, siendo el tamaño de dicho espacio de soluciones viables $\#S^* = n!$.

El BPP es un problema NP-duro que aparece frecuentemente en aplicaciones reales como el llenado de contenedores (o de palés) de transporte, la carga de camiones con restricciones de peso, o la creación de back-ups de archivos en medios electrónicos de almacenamiento. Las variantes principales se refieren al número de dimensiones consideradas al establecer la capacidad de los contenedores y el correspondiente tamaño de los objetos: BPP lineal o unidimensional, en el que la capacidad máxima *V* de los contenedores y los tamaños v_i de los *n* objetos a empaquetar son escalares referidos a una sola dimensión lineal, la cual puede ser longitud, peso, o coste (caso básico); BPP

multilineal, en el que la capacidad máxima de los contenedores y los tamaños de los *n* objetos se descomponen en *m* magnitudes unidimensionales no interrelacionadas geométricamente; BPP bidimensional (2D-BPP), en el que la capacidad máxima de los contenedores y los tamaños de los *n* objetos se interpretan como áreas que se descomponen en dos magnitudes unidimensionales interrelacionadas del espacio geométrico: longitud y anchura; BPP tridimensional (3D-BPP), en el que la capacidad máxima de los contenedores y los tamaños de los *n* objetos se interpretan como volúmenes que se descomponen en tres magnitudes unidimensionales interrelacionadas del espacio geométrico: longitud, anchura y altura.

4. *Problema de la partición*.

Definición (Problema de la partición). El problema de la partición (*Partition Problem*, PP) consiste en la partición de un conjunto *S* de *n* objetos, cada uno con un valor entero positivo v_i asociado, en dos subconjuntos S_1 y S_2 de forma que se minimice la diferencia en valor absoluto entre la suma de los valores de los objetos de S_1 y la suma de los valores de los objetos de S_2. Desde el punto de vista de la optimización combinatoria, el problema de la partición es una cuádrupla (*X*,*S*,*f*,*R*), donde:

- $X = \{x_1,...,x_{n+1}\}$, es el conjunto de variables del problema, en el que cada variable x_i, con $i \in \{1,...,n\}$, representa la posición *i* de la secuencia general de separación de los objetos y la variable x_{n+1} representa el número de objetos consecutivos de dicha secuencia a partir del primero que se asignan al subconjunto S_1 (los restantes $n - x_{n+1}$ objetos se asignan a S_2). El dominio finito común para las *n* variables x_1 a x_n es el conjunto de los *n* objetos a separar $D_1 = \{1,2,...,n\}$, y el dominio de la variable x_{n+1} es $D_2 = \{0,1,2,...,n\}$.

- $S = D_1^n \times D_2$, es el espacio de soluciones del problema, de tamaño $\#S = \#\left(D_1^n \times D_2\right) = n^n(n+1)$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1,\ldots,x_{n+1}) \in S$.

- $f : S \to \mathbb{R}$, es la función objetivo a minimizar, o diferencia en valor absoluto entre la suma de los valores de los objetos de S_1 y la suma de los valores de los objetos de S_2, determinada por:

$$f(\boldsymbol{x}) = f(x_1,\ldots,x_{n+1}) = \left| \sum_{i=1}^{x_{n+1}} v_{x_i} - \sum_{i=x_{n+1}+1}^{n} v_{x_i} \right|$$

  donde v_{x_i} es el entero positivo asociado al objeto x_i.

- $R = \{r_1,\ldots,r_{n-1}\}$, es el conjunto de restricciones:

$$r_i : \forall j \in \{1,\ldots,i\} : x_{i+1} \neq x_j \quad (i = 1,\ldots,n-1)$$

  Las restricciones r_1 a r_{n-1} que relacionan las variables x_1 a x_n aseguran que cada objeto se asigna una sola vez en la partición.

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1,\ldots,s_{n+1}) \in S$, que minimice globalmente la función objetivo f, de tal forma que se satisfagan todas las restricciones impuestas entre las variables. El espacio de soluciones viables es el conjunto $S^* = \{\boldsymbol{x} = (x_1,x_2,\ldots,x_{n+1}) \in S \mid \forall i \in \{1,\ldots,n-1\}, \forall j \in \{1,\ldots,i\} : x_{i+1} \neq x_j\}$, formado por las permutaciones de los n objetos para las variables x_1 a x_n, y cualquier valor en su dominio para la variable x_{n+1}, siendo el tamaño de dicho espacio de soluciones viables $\#S^* = n!(n+1)$.

El problema de la partición es un problema NP-duro que admite diversas versiones en las que se utilizan diferentes funciones objetivo, tal como el problema del corte mínimo (o máximo) de grafos, problema en el que se trata de separar los vértices de un grafo conexo en dos subconjuntos disjuntos tales que el número total de aristas entre vértices de diferentes subconjuntos sea mínimo (respectivamente, máximo). Esta versión del problema y especialmente su generalización, el k-corte mínimo de grafos o división del conjunto de vértices en k subconjuntos disjuntos que minimizan el número de interconexiones, aparece en aplicaciones reales tales como el diseño de circuitos electrónicos integrados.

5. *Problema de la asignación cuadrática*.

Definición (Problema de la asignación cuadrática). El problema de la asignación cuadrática (*Quadratic Assignment Problem*, QAP) consiste en la ubicación de *n* plantas industriales en *n* lugares posibles, teniendo en cuenta que entre cada dos plantas existe un flujo de transporte de materiales w_{ij} con un coste directamente proporcional a la distancia entre los lugares en que se hallen las plantas, de forma que se minimice el coste total del transporte de materiales entre las plantas. Desde el punto de vista de la optimización combinatoria, el problema de la asignación cuadrática es una cuádrupla (*X*,*S*,*f*,*R*), donde:

- $X = \{x_1,...,x_n\}$, es el conjunto de variables del problema, en el que cada variable x_i representa la planta industrial número *i* a ubicar. El dominio finito común para las *n* variables es el conjunto de los *n* lugares donde se deben ubicar las plantas $D = \{1,2,...,n\}$.

- $S = D^n$, es el espacio de soluciones del problema, de tamaño $\#S = \#D^n = n^n$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1,\ldots,x_n) \in S$.

- $f : S \to \mathbb{R}$, es la función objetivo a minimizar, o coste total del transporte de materiales entre las plantas, determinada por:

$$f(\boldsymbol{x}) = f(x_1,\ldots,x_n) = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} d(x_i, x_j)$$

  donde $d(x_i, x_j)$ es la distancia existente entre los lugares x_i y x_j, y w_{ij}, con $i, j \in \{1,...,n\}$, es la cantidad de material que debe transportarse de la planta número *i* a la planta número *j*. Intuitivamente, esta función de coste favorece la ubicación cercana de plantas entre las que exista un alto flujo de materiales.

- $R = \{r_1,... ,r_{n-1}\}$, es el conjunto de restricciones:

$$r_i : \forall j \in \{1,...,i\} : x_{i+1} \neq x_j \quad (i = 1,...,n-1)$$

Las restricciones r_1 a r_{n-1} que relacionan las variables x_1 a x_n aseguran que cada lugar es asignado una sola vez.

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1,...,s_n) \in S$, que minimice globalmente la función objetivo *f*, de tal forma que se satisfagan todas las restricciones impuestas entre las variables. El espacio de soluciones viables es el conjunto $S^* = \left\{ \boldsymbol{x} = (x_1,x_2,...,x_n) \in S \mid \forall i \in \{1,...,n-1\}, \forall j \in \{1,...,i\} : x_{i+1} \neq x_j \right\}$, formado por las permutaciones de los *n* lugares, siendo el tamaño de dicho espacio de soluciones viables $\#S^* = n!$.

El problema de la asignación cuadrática es NP-duro, ya que no se conoce ningún algoritmo de complejidad potencial que lo resuelva o que verifique si una solución es la óptima. De hecho, el problema del viajante comercial puede ser visto como un caso especial del QAP en el que los flujos de material entre plantas tienen el mismo valor constante y se producen únicamente en un sentido que relaciona a las plantas en forma de anillo. Además de la formulación original de ubicación de plantas industriales, el QAP es un modelo matemático para el problema de la ubicación de componentes electrónicos interconectados en una tarjeta de circuitos impresos o en un microchip, lo cual constituye una parte importante de la etapa diseño de circuitos digitales en la industria de la electrónica.

6. *Problema de la mochila*.

Definición (Problema de la mochila). El problema de la mochila (*Knapsack Problem*, KP) consiste en seleccionar de un conjunto de *n* objetos, cada uno con un valor v_i y un peso w_i asociados, el grupo de objetos que puede llevarse en una mochila, tales que su valor total sea máximo sin exceder un peso máximo *W*. Desde el punto de vista de la optimización combinatoria, el problema de la mochila es una cuádrupla (*X*,*S*,*f*,*R*), donde:

- $X = \{x_1,..., x_n\}$, es el conjunto de variables del problema, en el que cada variable x_i representa un objeto seleccionable. El dominio finito común

para las *n* variables es binario $D = \{0,1\}$, de manera que si la variable toma valor 1 indica que el objeto es seleccionado y si toma valor 0 el objeto no es seleccionado.

- $S = D^n$, es el espacio de soluciones del problema, de tamaño $\#S = \#D^n = 2^n$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1,\ldots,x_n) \in S$.
- $f : S \to \mathbb{R}$, es la función objetivo a maximizar, o valor total de los objetos seleccionados que se llevan en la mochila, función determinada por la expresión:

$$f(\boldsymbol{x}) = f(x_1,\ldots,x_n) = \sum_{i=1}^{n} v_i x_i$$

donde v_i es el valor del objeto *i*.

- $R = \{r\}$, es el conjunto de restricciones, formado únicamente por la restricción de peso máximo:

$$r : \sum_{i=1}^{n} w_i x_i \leq W$$

donde w_i es el peso del objeto *i*.

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1,\ldots,s_n) \in S$, que maximice globalmente la función objetivo *f*, de tal forma que se satisfaga la restricción de peso máximo. El espacio de soluciones *S* es el conjunto de las variaciones con repetición de *n* valores binarios, siendo el espacio de soluciones viables $S^* = \left\{ \boldsymbol{x} = (x_1,\ldots,x_n) \in S \mid \sum_{i=1}^{n} w_i x_i \leq W \right\}$ un subconjunto de *S* cuyo tamaño no puede determinarse directamente: $\#S^* < \#S = 2^n$.

El problema de la mochila es uno de los problemas clásicos de optimización combinatoria más sencillos de formalizar y entender, ha sido estudiado durante más de un siglo, y aunque se desconocen los orígenes de su nombre, ya se

menciona como tal en los tempranos trabajos del matemático Tobías Dantzig en el año 1930. Se trata de un problema NP-duro, al igual que el TSP, excepto si se puede realizar una división o partición de los objetos, ya que en ese caso existe un sencillo algoritmo exacto de complejidad potencial, consistente en ordenar los objetos de mayor a menor por ratio v_i/w_i (valor por unidad de peso) y a continuación tomar los k primeros objetos cuya suma de pesos sea justamente menor que el peso máximo W y añadir la parte del objeto k+1 necesaria para llegar a un peso total igual a W. El problema de la mochila (o una versión del mismo) aparece frecuentemente en entornos reales, tales como la selección de artículos a almacenar en un depósito, la selección de una cartera de inversiones financieras que maximicen la rentabilidad sin exceder un cuantía máxima invertida, e incluso la generación de claves criptográficas. Las variantes principales de problema de la mochila, las cuales pueden combinarse entre sí, son: KP multi-objetivo, en el que en vez de un solo objetivo a maximizar, el valor económico de los objetos seleccionados, puede haber otros objetivos de tipo medioambiental o social (caducidad, popularidad); KP multi-lineal, en el que en vez de una restricción unidimensional, el peso máximo permitido para los objetos seleccionados, puede haber otras características independientes (p.e., volumen) que limiten la selección de objetos; KP múltiple, en el que se consideran varias mochilas para el transporte de los objetos seleccionados.

7. *Problema de la máxima diversidad*.

Definición (Problema de la máxima diversidad). El problema de la máxima diversidad (*Maximum Diversity Problem*, MDP) consiste en seleccionar un subconjunto de m objetos de un conjunto total de n objetos, siendo $m < n$, con el fin de maximizar una función de distancia entre los objetos que mide la diversidad del subconjunto. Desde el punto de vista de la optimización combinatoria, el problema de la máxima diversidad es una cuádrupla (X,S,f,R), donde:

- $X = \{x_1,...,x_n\}$, es el conjunto de variables del problema, en el que cada variable x_i representa un objeto seleccionable. El dominio finito común

para las *n* variables es binario $D = \{0,1\}$, de manera que si la variable toma valor 1 indica que el objeto es seleccionado y si toma valor 0 el objeto no es seleccionado.

- $S = D^n$, es el espacio de soluciones del problema, de tamaño $\#S = \#D^n = 2^n$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1,\ldots,x_n) \in S$.
- $f : S \to \mathbb{R}$, es la función objetivo a maximizar, o distancia total entre los *m* objetos seleccionados que forman el subconjunto, función determinada por la expresión:

$$f(\boldsymbol{x}) = f(x_1,\ldots,x_n) = \sum_{i=1}^{n}\sum_{j=1}^{n} d(i,j)x_i x_j$$

donde $d(i,j)$ es la función de distancia que mide la diversidad entre los objetos *i* y *j*.

- $R = \{r\}$, es el conjunto de restricciones, formado únicamente por la restricción de número de objetos seleccionados:

$$r : \sum_{i=1}^{n} x_i = m$$

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1,\ldots,s_n) \in S$, que maximice globalmente la función objetivo *f*, de tal forma que se satisfaga la restricción de número de objetos seleccionados. El espacio de soluciones viables es el conjunto $S^* = \left\{ \boldsymbol{x} = (x_1,\ldots,x_n) \in S \mid \sum_{i=1}^{n} x_i = m \right\}$, formado por las variaciones con repetición de *n* valores binarios que suman *m*, equivalente al de las combinaciones de *n* objetos tomados de *m* en *m* (subconjuntos posibles de cardinal *m* de un conjunto de cardinal *n*), siendo el tamaño de dicho espacio de soluciones viables $\#S^* = \dfrac{n!}{m!(n-m)!}$.

El problema de la máxima diversidad es NP-duro, ya que no se conoce ningún algoritmo de resolución ni de verificación de complejidad potencial. Dentro de la teoría de grafos, el problema de la pandilla máxima (Maximum Clique Problem, MCP), que consiste en encontrar en un grafo simple *G* la mayor pandilla, es decir, el mayor subconjunto de vértices que forman un subgrafo completo de *G*, puede ser visto como un caso particular del MDP en el que los objetos son los vértices del grafo y la distancia entre vértices es binaria, igual a 1 si los vértices son adyacentes, e igual a 0 en caso contrario. Un ejemplo de aplicación del MDP aparece en la preservación de la biodiversidad cuando los recursos son limitados, y en otros contextos como la investigación médica, las políticas de gestión de personas, o la selección de carteras de inversión.

8. *Problema de la programación de operaciones de producción*.

Definición (Problema de la programación de operaciones de producción). El problema de la programación de operaciones de producción discreta (*Job-shop Scheduling Problem*, JSP) considera en su definición más básica una planta industrial dedicada a la fabricación de *p* piezas o artículos mediante un conjunto de *m* máquinas, de modo que cada máquina *j* es capaz de efectuar el proceso de producción de un subconjunto de los artículos, con un tiempo unitario de proceso diferente por artículo t_{jk}, el cual será nulo si la máquina *j* no puede producir el artículo *k*. El problema consiste en asignar máquina a un conjunto de *n* operaciones de producción, cada una dirigida a la fabricación de q_i unidades de un mismo artículo a_i, con el objetivo de minimizar el tiempo total de proceso de las operaciones. Desde el punto de vista de la optimización combinatoria, el problema de la programación de operaciones de producción es una cuádrupla (*X*,*S*,*f*,*R*), donde:

- $X = \{x_1,...,x_n\}$, es el conjunto de variables del problema, en el que cada variable x_i representa la operación número *i* a asignar. El dominio finito común para las *n* variables es el conjunto de las *m* máquinas donde se deben realizar las operaciones de producción $D = \{1,2,...,m\}$.

- $S = D^n$, es el espacio de soluciones del problema, de tamaño $\#S = \#D^n = m^n$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1,\ldots,x_n) \in S$.

- $f : S \to \mathbb{R}$, es la función objetivo a minimizar, o tiempo total de proceso (makespan) de las operaciones en las máquinas, determinada por:

$$f(\boldsymbol{x}) = f(x_1,\ldots,x_n) = \max\left\{ \sum_{i=1}^{n} q_i t_{x_i a_i} \delta(x_i, j) \mid j = 1,\ldots,m \right\}$$

  donde q_i es el número de piezas a fabricar en la operación i, $t_{x_i a_i}$ es el tiempo unitario de producción en la máquina asignada a la operación i del artículo a_i, y $\delta(x_i, j)$ es la función de Kronecker:

$$\delta(x_i, j) = \begin{cases} 0 & \text{si } x_i \neq j \\ 1 & \text{si } x_i = j \end{cases}$$

- $R = \{r_1,\ldots,r_n\}$, es el conjunto de restricciones:

$$r_i : t_{x_i a_i} \neq 0 \quad (i = 1,\ldots,n)$$

  Las restricciones r_1 a r_n aseguran que a cada operación i se le asigna una máquina capaz de realizar el proceso productivo del artículo correspondiente (tiempo de proceso de la máquina x_i para el artículo a_i no nulo).

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1,\ldots,s_n) \in S$, que minimice globalmente la función objetivo f, de tal forma que se satisfagan todas las restricciones impuestas entre las variables. El espacio de soluciones S es el conjunto de las variaciones con repetición de de orden n del conjunto de m máquinas, siendo el espacio de soluciones viables $S^* = \left\{ \boldsymbol{x} = (x_1, x_2,\ldots, x_n) \in S \mid \forall i \in \{1,\ldots,n\} : t_{x_i a_i} \neq 0 \right\}$ un subconjunto de S cuyo tamaño no puede determinarse directamente: $\#S^* < \#S = m^n$.

Como los anteriores, el JSP es un problema NP-duro para el que no se conocen algoritmos de resolución ni de verificación de complejidad potencial. El

JSP es uno de los problemas de optimización combinatoria que más variantes y generalizaciones admite, posiblemente más incluso que el VRP, debido a la enorme casuística que surge en los entornos de fabricación industrial. Las principales variantes del JSP son: JSP con planes de producción multiproceso de los artículos, en el que la fabricación de cada artículo se compone de varios procesos individuales que deben llevarse a cabo, normalmente en secuencia; JSP multi-objetivo, en el que además del tiempo total de proceso (o en su lugar) puede haber otros objetivos, como la reducción de costes de producción, la optimización del uso de las máquinas, la consideración de preferencias, la prioridad de las operaciones, etc.; JSP con fechas de lanzamiento y entrega de pedidos, en el que cada operación de producción tiene asociadas dos fechas críticas, la fecha de lanzamiento y la fecha de entrega; JSP con tiempos de preparación dependientes de la secuencia (sequence-dependent setup times) para las máquinas; JSP dinámico, en el que se consideran eventos o excepciones en tiempo real que pueden afectar a los programas de producción elaborados de manera que sea necesario realizar una reprogramación parcial de las operaciones. Además de las variantes principales anteriores, que además pueden recombinarse de forma arbitraria, existen muchas otras de mayor detalle, que también afectan a la definición del problema, ya que incrementan las variables consideradas, los objetivos de optimización y las restricciones a tener en cuenta, tales como la existencia de recursos auxiliares críticos (operarios, herramientas, utillaje, materiales), la incorporación de calendarios laborales, turnos de trabajo y planes de mantenimiento para las máquinas, la existencia de planes de proceso alternativos para la fabricación de los artículos y de planes de proceso con procesos que pueden realizarse en paralelo en lugar de secuencialmente, etc.

2.1.5. Resolución exacta de los problemas de optimización combinatoria: Búsqueda exhaustiva

Algoritmo de búsqueda exhaustiva

La resolución exacta de un problema de optimización combinatoria supone obtener la solución óptima global del problema. Debido a la complejidad algorítmica de los problemas de optimización combinatoria, los algoritmos exactos de resolución se basan en procedimientos de enumeración sistemática de soluciones o búsqueda exhaustiva.

Así, dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), con conjunto de variables $X = \{x_1,...,x_n\}$, de dominios discretos y finitos $D_i = \{1,2,...,p_i\}$, con $i = 1,2,...,n$, la estructura básica de un algoritmo de búsqueda exhaustiva es la siguiente:

1. Se genera la primera solución candidata del problema $\boldsymbol{x} = (1,1,\ldots,1) \in S$.
2. Si $\boldsymbol{x}$ verifica todas las restricciones *R* del problema, se almacena la solución en curso $\boldsymbol{x}$ como mejor solución obtenida hasta el momento por el algoritmo, $\boldsymbol{x}_{\text{opt}} := \boldsymbol{x}$.
3. Se genera la siguiente solución candidata del problema en la secuencia de valores de las variables dentro de sus dominios respectivos, $\boldsymbol{x} := \text{Siguiente}(\boldsymbol{x})$
4. Si la nueva solución en curso $\boldsymbol{x}$ verifica las restricciones *R* del problema, y es de mejor calidad que la mejor solución obtenida hasta el momento por el algoritmo, se almacena como tal, $\boldsymbol{x}_{\text{opt}} := \boldsymbol{x}$.
5. Si la solución en curso no es la última solución candidata posible del problema, $\boldsymbol{x} \neq (p_1,p_2,\ldots,p_n)$, se repiten los pasos 3 y 4. En caso contrario, el algoritmo termina, siendo la solución del problema la mejor solución obtenida por el algoritmo, $\boldsymbol{x}_{\text{opt}}$ (óptimo global).

Como se observa, el algoritmo de búsqueda exhaustiva asegura la obtención de la solución óptima global del problema gracias a la enumeración sistemática de todas las soluciones candidatas. Por ello, el número de iteraciones que el algoritmo debe efectuar es igual al tamaño del espacio de soluciones:

$$\#S = \#\left(\bigtimes_{i=1}^{n} D_i \right) = \prod_{i=1}^{n} p_i = p_1 \cdot p_2 \cdots p_n$$

En ocasiones, para determinados problemas es posible encontrar un procedimiento que genere directamente la secuencia ordenada de soluciones candidatas viables, por ejemplo, cuando las soluciones viables sean variaciones, permutaciones o combinaciones, lo que evita generar todas las soluciones candidatas y comprobar para cada una las restricciones del problema. En tales casos, el algoritmo de búsqueda exhaustiva se simplifica algo, si bien el número de iteraciones que debe realizar sigue siendo muy elevado, igual al tamaño del espacio de soluciones viables $\#S^*$.

Limitaciones de los algoritmos exactos de optimización combinatoria

Los algoritmos exactos de optimización combinatoria basados en la búsqueda exhaustiva determinan la solución óptima global del problema gracias a la enumeración sistemática de soluciones. Sin embargo, la complejidad algorítmica de la búsqueda exhaustiva en el caso peor (enumeración de todas las soluciones candidatas) es, como se ha comprobado, potencial-exponencial, ya que el número de iteraciones que se deben realizar para un problema de *n* variables con dominios $D_i = \{1,2,...,p_i\}$, siendo $i = 1,2,...,n$, es igual a $\#S = p_1 \cdot p_2 \cdots p_n$. Incluso en casos mejores (enumeración de permutaciones o combinaciones como soluciones candidatas viables) la complejidad del algoritmo de búsqueda exhaustiva es exponencial o superior.

Por tanto, los enfoques exactos que rastrean todo el espacio de soluciones para encontrar siempre la solución óptima global son válidos tan solo para instancias de tamaño muy reducido de los problemas de optimización. En caso

contrario, los tiempos de ejecución de estos métodos son irrealizables en la práctica. No obstante, los algoritmos exactos son de gran importancia desde un punto de vista teórico, pues permiten determinar la solución óptima global de instancias de pequeño tamaño de los problemas, que pueden servir para el análisis y la evaluación de la eficiencia de otros métodos no exactos (algoritmos aproximados).

2.1.6. Resolución aproximada de los problemas de optimización combinatoria: Heurísticas y meta-heurísticas

La inviabilidad práctica de los métodos exactos de resolución de problemas reales de optimización combinatoria ha hecho necesaria la búsqueda de métodos aproximados que permitan obtener soluciones óptimas locales de calidad en un tiempo factible para este tipo de problemas. El diseño de métodos algorítmicos aproximados ha tenido especial relevancia en las últimas décadas, y se ha visto favorecido por los avances en potencia de cálculo de los computadores. Las técnicas algorítmicas de resolución más utilizadas son las *heurísticas* (algoritmos heurísticos) y las *meta-heurísticas* (algoritmos meta-heurísticos). A continuación se describe de forma general en qué consiste cada una de estas técnicas y cuáles son sus características principales.

Heurísticas

Un algoritmo de optimización heurístico, o simplemente heurística[1], es una técnica que trata de encontrar solución a un problema de optimización haciendo uso del conocimiento específico existente sobre dicho problema. Una posible definición de algoritmo heurístico es la aportada por A. Díaz [DIAZ96]:

> "*Un método heurístico es un procedimiento para resolver un problema de optimización bien definido mediante una aproximación intuitiva, en la que la estructura del problema se utiliza de forma inteligente para obtener una buena solución.*"

[1] El término heurística proviene del verbo griego *heuriskein,* cuyo significado es encontrar o descubrir.

Otra definición similar es la propuesta por H. Zanakis et al. [ZANA81]:

> "*Una heurística es un procedimiento simple, basado en el sentido común, que se supone obtiene una buena solución (no necesariamente óptima) a problemas difíciles de un modo sencillo y rápido.*"

Una heurística utiliza información concreta y características conocidas del problema en la exploración del espacio de soluciones de manera que la búsqueda se intensifica dentro de las áreas que sean más prometedoras, con el objetivo de encontrar una buena solución de forma rápida.

Obviamente, el diseño y análisis de algoritmos heurísticos está completamente ligado al tipo de problema a resolver, por lo que su estudio constituye un área muy amplia a tratar, que se divide según los problemas o tipos de problemas considerados. Incluso puede hablarse del nivel de especificidad de una heurística con respecto a los problemas a los que puede aplicarse:

- Heurísticas muy específicas, aplicables sólo a ciertas instancias de un problema de optimización. Por ejemplo, una heurística que resulte útil para resolver instancias del problema del viajante en las que los lugares a visitar se encuentren agrupados en conglomerados claramente diferenciados.

- Heurísticas intermedias, aplicables a cualquier instancia de un problema de optimización. Por ejemplo, una heurística que sea útil para resolver cualquier instancia del problema del viajante.

- Heurísticas generales, aplicables a cualquier problema de optimización dentro de un tipo concreto. Por ejemplo, una heurística que sea útil para resolver problemas de planificación de rutas, tales como el problema del viajante o el problema de la asignación de rutas a vehículos.

Los algoritmos heurísticos suelen ser muy potentes y útiles en su aplicación práctica, pero para su diseño se requiere tener un conocimiento muy preciso de las características del problema y de su espacio de soluciones. Generalmente,

las heurísticas más eficientes se desarrollan para la resolución de problemas de tipo académico con una formulación sencilla y bien conocidos, ya que la dificultad de encontrar heurísticas adaptables a problemas reales más complejos, con función objetivo y restricciones complejas, resulta difícil o incluso imposible desde un punto de vista práctico por cuestiones de tiempo.

Meta-heurísticas

A principio de la década de los ochenta comenzaron a desarrollarse una serie de métodos, denominados *meta-heurísticos*, es decir, heurísticos de nivel superior o más generales, que intentan superar las limitaciones de eficiencia y especificidad de las heurísticas tradicionales mediante una exploración mayor del espacio de soluciones, lo cual ha sido factible gracias al incremento progresivo en potencia de cálculo de los computadores.

Existen varias interpretaciones del concepto de meta-heurística. Algunos autores definen las meta-heurísticas como estructuras algorítmicas de alto nivel que guían otras heurísticas de bajo nivel con el fin de obtener mejores resultados. Así, F. Glover [GLOV86] definió estos métodos de la siguiente forma:

> *"Una meta-heurística es una estrategia maestra de alto nivel que guía y modifica otras heurísticas para producir soluciones mejores que las que se generan normalmente en procedimientos de optimización local."*

Otros autores definen las meta-heurísticas simplemente como evoluciones complejas de las heurísticas iniciales, es decir, como heurísticas complejas modernas. Así, J.P. Kelly et al [KELL96] aporta la siguiente definición:

> *"Las meta-heurísticas son métodos aproximados diseñados para resolver problemas de optimización combinatoria en los que las heurísticas clásicas no son efectivas. Proporcionan una estructura general para crear nuevos algoritmos híbridos combinando*

> *diferentes conceptos derivados de la inteligencia artificial, la evolución biológica, y los métodos estadísticos."*

La definición de meta-heurística que se va a considerar aquí es la derivada directamente del propio prefijo *meta* del término: una meta-heurística es una heurística completamente general, es decir, una heurística aplicable a cualquier problema de optimización. Por tanto, *un algoritmo meta-heurístico de optimización, o meta-heurística, es una técnica aproximada de alto nivel que busca solución a un problema de optimización utilizando en el proceso de búsqueda sólo conocimiento general común a todos los problemas de optimización*.

Las meta-heurísticas puras pueden utilizarse en cualquier tipo de problema de optimización, ya que basan la búsqueda sólo en la información común a todos los problemas de optimización (sucintamente, la existencia de una función objetivo numérica de variables discretas y un conjunto de restricciones numéricas entre las variables), y restringen el uso de la información específica del problema particular a la formulación matemática, y a la codificación de las soluciones de forma que se asegure la viabilidad de las mismas respecto a las restricciones impuestas. Para ello, las meta-heurísticas exploran un área mucho mayor del espacio de soluciones que las heurísticas específicas, con el objetivo de encontrar una buena solución en tiempo factible. Por esta razón, estas técnicas son más apropiadas para su aplicación a problemas reales con una formulación compleja, ya que no necesitan utilizar información específica en la exploración del espacio de soluciones viables del problema. Se trata de algoritmos que pueden ser comparativamente menos eficientes que los heurísticos específicos del problema, pero que pueden aplicarse con poco esfuerzo a problemas complejos para los que no existan o no se conozcan heurísticas apropiadas.

Esta definición de meta-heurística no está en conflicto con las anteriores. En efecto, las meta-heurísticas pueden también utilizarse en la resolución de un problema de optimización concreto como estructura o marco algorítmico en combinación con heurísticas específicas de más bajo nivel adaptadas al problema, formando lo que podría interpretarse como un algoritmo híbrido

multinivel, con el objetivo de conseguir una mayor eficiencia resolutiva. De hecho, la combinación de técnicas heurísticas y meta-heurísticas es muy común en la investigación y el desarrollo de aplicaciones prácticas.

Precisamente, este trabajo de investigación está orientado al estudio de las meta-heurísticas como algoritmos generales de optimización combinatoria. Concretamente se centra en el análisis detallado y la mejora de una de las meta-heurísticas fundamentales: el *recocido simulado*. Además del recocido simulado existe un gran número de meta-heurísticas en la literatura científica: *búsqueda tabú*, *algoritmos evolutivos y genéticos*, *optimización por enjambre de partículas inteligentes*, *búsqueda dispersa*, *algoritmo GRASP*, *optimización por colonia de hormigas*, etc. Para una mejor compresión y enmarque del recocido simulado dentro del conjunto de las meta-heurísticas, en los siguientes apartados se realiza una breve descripción de las meta-heurísticas más importantes. Como paso previo se establece una taxonomía o clasificación básica de las meta-heurísticas de acuerdo con diversos criterios.

Tipos de algoritmos aproximados

Los algoritmos aproximados (heurísticos y meta-heurísticos) que se emplean en la resolución de los problemas de optimización combinatoria se pueden clasificar según diferentes criterios:

- *Algoritmos transformativos* y *algoritmos constructivos*. Los algoritmos transformativos son aquellos en los se parte de una o varias soluciones candidatas del problema, completas y viables respecto a las restricciones, que van transformándose parcialmente según avanza el proceso de optimización. Los algoritmos constructivos son aquellos en los que se parte de una o varias soluciones parciales viables del problema, llamadas soluciones semilla, que van completándose paso a paso mediante un proceso constructivo en el que se asignan secuencialmente valores a las variables indeterminadas de acuerdo con algún criterio de optimización, de modo que en cada paso se cumplan las restricciones del problema con respecto a las variables ya asignadas.

- *Algoritmos de búsqueda simple* o *trayectoriales* y *algoritmos poblacionales*. En los algoritmos de búsqueda simple se considera una única solución en curso, que se transforma, o construye, en el tiempo, formando una trayectoria por el espacio de soluciones del problema. En los algoritmos de búsqueda múltiple o poblacionales se contemplan simultáneamente varias soluciones en curso, las cuales se van transformando, o se construyen, en el tiempo, estableciéndose entre ellas una cierta interacción.

- *Algoritmos deterministas* y *algoritmos aleatorios*. Un algoritmo determinista es aquel que, para una instancia concreta de un problema de optimización y ante una misma solución inicial, ya sea completa como en los algoritmos transformativos, o parcial como en los algoritmos constructivos, obtiene en cada ejecución siempre la misma solución óptima final del problema. Por el contrario, un algoritmo aleatorio es aquel que presenta algún componente de azar o probabilístico en su operación, de tal modo que, para una instancia concreta de un problema de optimización y ante una misma solución inicial, obtiene en diferentes ejecuciones distintas soluciones óptimas finales del problema.

- *Algoritmos estáticos* y *algoritmos dinámicos*. Los algoritmos estáticos son aquellos que no modifican durante toda la ejecución sus parámetros y estructuras de funcionamiento, tales como parámetros de aleatorización, estructuras de vecindad, criterios de aceptación de soluciones, o criterios de terminación. Por el contrario, los algoritmos dinámicos son los que modifican dinámicamente durante la ejecución sus parámetros y estructuras operativas, por ejemplo, reduciendo progresivamente la perturbación aleatoria, cambiando de estructura de vecindad, o alterando el criterio de aceptación de soluciones o el criterio de terminación. Un caso especial de algoritmos dinámicos son aquellos que modifican la función objetivo del problema o las restricciones, aunque en realidad en estos casos el dinamismo suele ser más una característica del propio problema a resolver que del algoritmo en sí.

- *Algoritmos integrales* y *algoritmos de división*. Un algoritmo integral es aquel que considera el problema de optimización como un todo y obtiene la mejor solución posible considerando globalmente el conjunto de variables, la función objetivo y el conjunto de restricciones originales en cada fase del proceso de optimización. Un algoritmo de división es aquel que divide el problema en partes o subproblemas, por ejemplo, separando el conjunto de variables del problema en subconjuntos disjuntos de variables, cada uno con una función objetivo y un conjunto de restricciones particulares, derivadas del objetivo y las restricciones del problema original. Estos algoritmos optimizan por separado cada subproblema para al final integrar las soluciones de los subproblemas en una solución global.

Diversificación (exploración) e intensificación (explotación) de un algoritmo aproximado

La diversificación y la intensificación son cualidades de los algoritmos aproximados, y más específicamente de las meta-heurísticas, que fueron desarrolladas originalmente en el ámbito de la búsqueda tabú [GLOV86]. Constituyen una medida de la capacidad o la potencialidad de las meta-heurísticas:

- La *diversificación*, también llamada *capacidad de exploración*, es la cualidad de un algoritmo aproximado de explorar sin exhaustividad una amplia región del espacio de soluciones, visitando de forma más o menos aleatoria puntos del espacio (soluciones) alejados entre sí. La diversificación tiene como objetivo escapar de los óptimos locales propios de las búsquedas sobre pequeñas regiones del espacio. Los algoritmos aproximados que potencian la diversificación se caracterizan por encontrar mejores soluciones que los algoritmos basados en la optimización local, pero más lentamente, es decir, suelen ser de convergencia lenta.

- La *intensificación*, también conocida como *capacidad de explotación*, es la cualidad de un algoritmo aproximado de explorar exhaustivamente o de forma intensa una pequeña región del espacio de soluciones, visitando la totalidad o una gran mayoría de los puntos del espacio (soluciones) de dicha región. La intensificación tiene como objetivo encontrar la mejor solución posible dentro de una vecindad, y es útil sobre todo cuando se realiza en regiones del espacio que sean prometedoras. Obviamente, para determinar esas regiones más prometedoras del espacio es conveniente utilizar información concreta sobre el problema y su espacio de soluciones, lo cual precisamente constituye la base de los algoritmos heurísticos. Los algoritmos aproximados que potencian la intensificación se caracterizan por encontrar buenas soluciones de forma rápida, es decir, suelen ser rápidamente convergentes.

Taxonomía de los principales algoritmos aproximados meta-heurísticos

Teniendo en cuenta los criterios de clasificación anteriores puede establecerse una taxonomía jerárquica de las meta-heurísticas fundamentales. Dado que los cinco criterios establecidos para los algoritmos aproximados no son excluyentes entre sí, para evitar formar una compleja red de meta-heurísticas, en esta clasificación se ha considerado un orden de importancia en los criterios:

1. Modo de generación de soluciones (algoritmos transformativos y algoritmos constructivos).
2. Número de soluciones en curso (algoritmos de búsqueda simple y algoritmos poblacionales).
3. Determinismo (algoritmos deterministas y algoritmos aleatorios).
4. Adaptabilidad (algoritmos estáticos y algoritmos dinámicos).
5. Integralidad (algoritmos integrales y algoritmos de división).

La tabla 2.2 ofrece una lista de las principales meta-heurísticas de optimización combinatoria.

			META-HEURÍSTICAS
Algoritmos transformativos	**Búsqueda simple**		Búsqueda local (*Local Search*, LS)
			Búsqueda de vecindad variable (*Variable Neighborhood Search*, VNS)
			Búsqueda local iterativa (*Iterated Local Search*, ILS)
			Recocido simulado (*Simulated Annealing*, SA)
			Búsqueda tabú (*Tabu Search*, TS)
	Poblacionales	**Uni-poblacionales**	Algoritmos genéticos (*Genetic Algorithms*, GA)
			Algoritmos meméticos (*Memetic Algorithms*, MA)
			Algoritmos culturales (*Cultural Algorithms*, CA)
			Optimización por enjambre de partículas (*Particle Swarm Optimization*, PSO)
			Colonia de abejas artificiales, (*Artificial Bee Colony*, ABC)
			Búsqueda dispersa (*Scatter Search*, SS)
			Reencadenamiento de trayectorias (*Path Relinking*, PR)
		Multi-poblacionales	Algoritmos genéticos paralelos (*Parallel Genetic Algorithms*, PGA)
			Optimización por enjambre múltiple de partículas (*Multiple Particle Swarm Optimization*, MPSO)
			Colonia múltiple de abejas artificiales, (*Multiple Artificial Bee Colony*, MABC)

Tabla 2.2: Principales meta-heurísticas de optimización combinatoria

Algoritmos constructivos	**Búsqueda simple**		Algoritmos voraces simples (*Simple Greedy Algorithms*, SGA)
			GRASP - Procedimiento de búsqueda voraz adaptativa aleatorizada (*Greedy Randomized Adaptive Search Procedure*, GRASP)
			POPMUSIC - Meta-heurística de optimización parcial bajo condiciones especiales de intensificación (*Partial Optimization Meta-heuristic Under Special Intensification Conditions*, POPMUSIC)
	Poblacionales	**Uni-poblacionales**	Optimización por colonia de hormigas (*Ant Colony Optimization*, ACO)
		Multi-poblacionales	Optimización por equipos asíncronos (*Asynchronous Teams Optimization*, ATO)

Tabla 2.2 (cont.): Principales meta-heurísticas de optimización combinatoria

La aceptación y desarrollo – por parte de los investigadores – de procedimientos heurísticos y meta-heurísticos para la resolución aproximada de problemas de optimización se refleja en:

- El gran números de libros publicados en este campo [ACKL87], [AART89], [REEE93], [REEV95], [RAYW95], [DIAZ96], [MICH00], [REEE02], [GLOV03],
- revistas especializadas: *Journal of Heuristics*, *Computational Optimization and Applications, Journal of Combinatorial Optimization, Computers and Operations Research, Evolutionary Computation, European Journal of Operational Research, INFORMS Journal on Computing, Inteligencia artificial*,
- artículos de investigación [LOCA00], [ALVA03], [DIAZ03], [DIAZ08],
- congresos: *European Conference on Evolutionary Computational,* el *Metaheuristics International Conference* o el *Congreso Español de Metaheurísticas y Algoritmos Evolutivos y Bioinspirados*,

- asociaciones como *HEUR* (http://heur.uv.es) o *Metaheuristics Network* (www.metaheuristics.org) con financiación de la Unión Europea, que aportan numerosa y valiosa información en Internet.

Actualmente existe un crecimiento espectacular de los algoritmos meta-heurísticos. Algunos de estos métodos se han desarrollado para la resolución de problemas concretos sin poder hacerse una generalización para otros problemas. Todas las meta-heurísticas básicas mencionadas en la tabla anterior se describen brevemente en apartados posteriores (2.2 a 2.9), haciendo especial énfasis en las meta-heurísticas transformativas de búsqueda simple y sus variantes, grupo al que pertenece el recocido simulado, objeto principal de estudio en la investigación de la tesis doctoral.

2.2. ALGORITMOS TRANSFORMATIVOS DE BÚSQUEDA SIMPLE I: BÚSQUEDA LOCAL

2.2.1. Búsqueda local

Los algoritmos de búsqueda local (*Local Search*, SL) [AART97] constituyen una clase interesante de algoritmos generales de optimización aproximada transformativos y de búsqueda simple o trayectoriales, basados en la técnica básica de *generación y prueba*. Se trata de un método iterativo simple que permite encontrar soluciones aproximadas rápidamente. De hecho, la búsqueda local es la base de muchos de los algoritmos aproximados usados en problemas de optimización combinatoria.

En esencia, el algoritmo comienza generando una solución arbitraria completa que verifique las restricciones del problema. A partir de entonces se van generando secuencialmente nuevas soluciones vecinas que verifiquen las restricciones. Para cada nueva solución generada se comprueba si mejora la calidad frente a la solución en curso, de manera que en caso afirmativo la

nueva solución mejor pasa a ser la solución en curso. El algoritmo termina cuando se ha explorado totalmente (o con probabilidad cercana a 1) el conjunto de soluciones vecinas a la actual sin que se produzcan mejoras cualitativas respecto a ésta. La solución del problema es precisamente esta solución en curso, la cual será, por definición, un óptimo local del problema.

Existen diferentes formas de dirigir la búsqueda local. Por lo general, la complejidad computacional de un procedimiento de búsqueda local depende del tamaño de la vecindad. Cuanto mayor sea la vecindad, mayor será el tiempo necesario para examinarla, y mejores las soluciones óptimas locales obtenidas como resultado.

Dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), la estructura básica de un algoritmo de búsqueda local es la siguiente:

1. Se establece como solución inicial en curso una solución candidata viable, elegida al azar o de forma heurística, $\boldsymbol{x} := \boldsymbol{x}_0$.
2. Mediante una función generatriz de soluciones vecinas viables *g*, se obtiene a partir de ***x*** una nueva solución candidata ***y***, $\boldsymbol{y} := g(\boldsymbol{x})$.
3. Si la nueva solución ***y*** es de mejor calidad que la actual ***x***, $f(\boldsymbol{y}) < f(\boldsymbol{x})$ en caso de minimización o $f(\boldsymbol{y}) > f(\boldsymbol{x})$ en caso de maximización, entonces la nueva solución ***y*** pasa a ser la solución en curso, $\boldsymbol{x} := \boldsymbol{y}$.
4. Si existen soluciones viables en la vecindad de la actual aún no procesadas, se reiteran los pasos 2 y 3. En caso contrario, si no puede encontrarse una solución de mejor calidad dentro de la vecindad de la actual, el algoritmo termina. La solución del problema es la última solución en curso ***x*** del procedimiento (óptimo local).

La función generatriz de soluciones vecinas viables *F* puede ser determinista o aleatoria. En el primer caso (búsqueda local determinista), la vecindad asociada a una solución en curso se recorre de forma secuencial, hasta que la solución en curso sea sustituida o se hayan considerado todas las soluciones vecinas a la actual y el algoritmo termine. Por el contrario, si la función

generatriz de soluciones vecinas viables es aleatoria (búsqueda local aleatoria), la estructura de vecindad asociada a una solución en curso se recorre al azar, hasta que la solución actual sea sustituida o se llegue a un límite máximo de iteraciones y el algoritmo termine. En general, este límite máximo de iteraciones debe implicar (con probabilidad cercana a 1) que en el recorrido aleatorio se han explorado todas las soluciones vecinas a la solución en curso.

A pesar de su sencillez, generalidad y flexibilidad, los algoritmos de búsqueda local presentan el inconveniente de que obtienen soluciones óptimas locales de calidad limitada. La calidad de la solución obtenida por un algoritmo de búsqueda local depende directamente de la solución inicial elegida y del tamaño de la vecindad considerada. Además, el carácter local de la búsqueda hace que la calidad media de las soluciones con respecto a la óptima empeore a medida que crece el tamaño del problema [JOHN85].

Para mejorar la eficiencia del algoritmo se puede optar por ejecutarlo un gran número de veces con diferentes soluciones iniciales factibles, guardando la mejor solución que se haya obtenido en las diferentes búsquedas. Sin embargo, obviamente este procedimiento de mejora tampoco garantiza que se encuentre el óptimo global.

2.2.2. Búsqueda de vecindad variable

La búsqueda de vecindad variable (*Variable Neighborhood Search*, VNS) es una meta-heurística de optimización combinatoria propuesta por Mladenovìc y Hansen [MLAD97] en 1997, que intenta reducir el problema de la convergencia en óptimos locales cambiando sistemáticamente la estructura de vecindad durante un procedimiento de búsqueda local [HANS01], [HANS02], [HANS03].

Con respecto a su funcionamiento, esta técnica toma como base un conjunto de p estructuras de vecindad alternativas, g_k, con $k = 1,..., p$, a utilizar durante el proceso de búsqueda. Cada una de estas estructuras de vecindad g_k se construye con una métrica diferente (medida de la distancia entre soluciones) y/o respecto a una función de transformación simple diferente, de forma que

una solución óptima local respecto a una vecindad g_k no tiene por qué ser, en general, un óptimo local respecto a otra vecindad $g_{k'}$. Esto significa que la topología del espacio de búsqueda queda determinada por la estructura de vecindad elegida. Las implicaciones de este principio son claras: si a partir de una solución inicial $\boldsymbol{x}_0$ se aplica un procedimiento de búsqueda local con una vecindad g_k, se obtendrá una solución óptima local $\boldsymbol{x}_1$, y si a continuación a partir de ésta solución $\boldsymbol{x}_1$ se aplica de nuevo un procedimiento de búsqueda local con otra estructura de vecindad $g_{k'}$, podrá obtenerse un segundo óptimo local $\boldsymbol{x}_2$, que mejorará el anterior.

El principio básico de la técnica VNS, esto es, el cambio sistemático de estructura de vecindad durante un procedimiento de búsqueda local, puede implementarse de diferentes formas, dando lugar a diversas variantes de la búsqueda de vecindad variable. La más representativa es la búsqueda básica de vecindad variable (*Basic Variable Neighborhood Search*, BVNS), de carácter mixto determinista-aleatorio. Básicamente, el algoritmo BVNS comienza generando una solución arbitraria completa que verifique las restricciones del problema. Tomando como base la primera estructura de vecindad del conjunto de vecindades definido, se ejecuta una búsqueda local aleatoria completa con el objetivo de alcanzar una solución óptima local. A continuación, se comprueba si la solución óptima local obtenida mejora la calidad frente a la solución en curso. En caso afirmativo, la nueva solución óptima local pasa a ser la solución en curso, y a continuación se repite el proceso de mejora por búsqueda local aleatoria, pero tomando de nuevo como base la primera estructura de vecindad del conjunto de vecindades. En caso contrario, se mantiene la solución en curso, y se repite el proceso de mejora por búsqueda local con la siguiente estructura de vecindad del conjunto de vecindades. El algoritmo termina cuando se hayan considerado todas las estructuras de vecindad sin que se produzcan mejoras cualitativas respecto a la solución en curso. Esta solución en curso constituye la solución del problema, la cual será, por definición, un óptimo local.

Dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), la estructura básica del algoritmo de búsqueda básica de vecindad variable es la siguiente:

1. Se define un conjunto de *p* estructuras de vecindad alternativas, g_k, con $k = 1,...,p$.

2. Se establece como solución inicial en curso una solución candidata viable, elegida al azar o de forma heurística, $\boldsymbol{x} := \boldsymbol{x}_0$.

3. Se toma como base la primera estructura de vecindad del conjunto de vecindades, $k := 1$.

4. Se ejecuta un procedimiento de búqueda local aleatoria para encontrar un óptimo local respecto a la vecindad g_k, $\boldsymbol{y} := \text{BúsquedaLocal}(\boldsymbol{x}, g_k)$

5. Si la solución encontrada ***y*** es mejor que la actual ***x***, $f(\boldsymbol{y}) < f(\boldsymbol{x})$ en caso de minimización o $f(\boldsymbol{y}) > f(\boldsymbol{x})$ en caso de maximización, entonces la nueva solución ***y*** pasa a ser la solución en curso, $\boldsymbol{x} := \boldsymbol{y}$, y se toma de nuevo como base la primera estructura de vecindad del conjunto de vecindades, $k := 1$. En caso contrario, se mantiene la solución actual ***x***, y se considera la siguiente estructura de vecindad del conjunto de vecindades, $k := k + 1$.

6. Si $k \leq p$, se repiten de nuevo los pasos 4 y 5. En caso contrario, el algoritmo termina, siendo la solución del problema la última solución en curso ***x*** del proceso (óptimo local de las *p* estructuras de vecindad).

Otras variantes importantes de la búsqueda de vecindad variable son:

- La búsqueda de vecindad variable descendente, (*Variable Neighborhood Descent*, VND), de carácter determinista y estructura algorítmica igual a la del BVNS, en la que se van eligiendo secuencialmente las estructuras de vecindad del conjunto de vecindades, pero se emplea un procedimiento de búsqueda local determinista para encontrar el óptimo local respecto a cada vecindad.

- La búsqueda reducida de vecindad variable (*Reduced Variable Neighborhood Search*, RVNS), de carácter aleatorio, consistente en un único procedimiento de búsqueda local aleatoria en el que tras cada mejora de la solución en curso se produce un cambio aleatorio de estructura de vecindad. En este caso, el criterio de terminación del algoritmo es un límite máximo de iteraciones sin mejora en la calidad de las nuevas soluciones respecto a la actual. Se trata de un límite arbitrario que puede elegirse de forma similar al caso de la búsqueda local aleatoria, por ejemplo, un límite que implique (con probabilidad cercana a 1) que en el recorrido aleatorio se han explorado todas las soluciones vecinas a la solución en curso respecto a la estructura de vecindad actual.

Finalmente, pueden citarse otras variantes más complejas de la búsqueda de vecindad variable, tales como la búsqueda descompuesta de vecindad variable (*Variable Neighborhood Decomposition Search*, VNDS), la búsqueda sesgada de vecindad variable (*Skewed Variable Neighborhood Search*, SVNS), o la búsqueda paralela de vecindad variable (*Parallel Variable Neighborhood Search*, PVNS) [HANS10].

2.2.3. Búsqueda local iterativa

La búsqueda local iterativa (*Iterated Local Search*, ILS) [STUT99], es una meta-heurística que utiliza iterativamente una estrategia de búsqueda local en dos niveles capaz de generar una sucesión convergente de óptimos locales del problema. Esta técnica ha sido redescubierta varias veces en la literatura con diferentes nombres: *descenso iterativo* (*Iterated Descent*, ID), *optimización local encadenada* (*Chained Local Optimization*, CLO). Los primeros que la aplicaron fueron Lin y Kernigham para la resolución del problema del viajante de comercio, TSP [LINK73]. Desde entonces este método ha sido utilizado numerosas veces en distintos problemas de optimización, tal como la programación de operaciones [RUIZ05].

La idea fundamental de la búsqueda local iterativa consiste en obtener la solución de un problema de optimización combinatoria (*X*,*S*,*f*,*R*) mediante una estrategia de búsqueda local de dos niveles: un procedimiento de búsqueda local determinista embebido o de bajo nivel que actúa directamente sobre el espacio de soluciones *S* y genera iterativamente un conjunto de óptimos locales del problema, y un segundo procedimiento de búsqueda local de alto nivel que actúa sobre el subespacio $S_{opt} \subseteq S$ de los óptimos locales obtenidos al aplicar el primer procedimiento [LOUR01].

En esencia, la búsqueda local iterativa comienza generando una solución arbitraria completa que verifique las restricciones del problema. A partir de esta solución inicial se ejecuta una búsqueda local determinista completa sobre una estructura de vecindad concreta, con el objetivo de alcanzar una solución óptima local, la cual constituye la primera solución en curso del algoritmo. En este punto, se inicia la parte iterativa del algoritmo, aplicándose sucesivamente las tres operaciones siguientes:

- Una alteración o perturbación de la solución en curso, que tiene por objetivo formar una nueva solución fuera de la vecindad de la solución en curso (óptimo local). Esta perturbación debe tener carácter aleatorio y su objetivo es salir fuera del entorno del óptimo local actual y explorar otras regiones del espacio de soluciones. La entidad de la perturbación tiene que definirse con detalle para evitar que, si es demasiado grande, la búsqueda local iterativa se transforme en un algoritmo de búsqueda aleatoria, o que, si es demasiado pequeña, pueda no ser suficiente para salir del entorno del óptimo local.

- Un procedimiento de búsqueda local determinista de bajo nivel que parte de la solución alterada y tiene por objetivo encontrar en su vecindad un nuevo óptimo local. Este procedimiento suele ser el mismo que el utilizado en la generación de la primera solución en curso.

- Un método de aceptación que permite decidir entre la solución en curso (óptimo local previo a la perturbación) y el nuevo óptimo local, cuál será la nueva solución en curso.

Como se observa, las tres operaciones de la parte iterativa del algoritmo forman a su vez un procedimiento de búsqueda local aleatoria de alto nivel sobre el subespacio de óptimos locales obtenidos mediante la búsqueda local determinista de bajo nivel.

En consecuencia, dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), la estructura básica de un algoritmo de búsqueda local iterativa es la siguiente:

1. Se genera, al azar o de forma heurística, una solución candidata viable $\boldsymbol{x}_0$.
2. Partiendo de la solución $\boldsymbol{x}_0$, se ejecuta un procedimiento de búsqueda local determinista utilizando una estructura de vecindad *g*, con el fin de encontrar un óptimo local, que constituye la primera solución en curso del algoritmo, $\boldsymbol{x} := \text{BúsquedaLocal}(\boldsymbol{x}_0, g)$.
3. Se almacena la solución en curso $\boldsymbol{x}$ como mejor solución obtenida hasta el momento por el algoritmo, $\boldsymbol{x}_{\text{opt}} := \boldsymbol{x}$.
4. Se aplica una perturbación aleatoria a la solución en curso para transformarla en otra solución fuera de su vecindad, $\boldsymbol{x}' := \text{Perturbación}(\boldsymbol{x})$
5. Partiendo de la solución alterada $\boldsymbol{x}'$, se ejecuta un procedimiento de búsqueda local determinista utilizando la etructura de vecindad *g*, con el fin de encontrar un nuevo óptimo local, $\boldsymbol{y} := \text{BúsquedaLocal}(\boldsymbol{x}', g)$.
6. Si la nueva solución $\boldsymbol{y}$ cumple el criterio de aceptación respecto a la solución en curso $\boldsymbol{x}$, entonces la nueva solución $\boldsymbol{y}$ pasa a ser la solución en curso, $\boldsymbol{x} := \boldsymbol{y}$. En tal caso, si la nueva solución en curso $\boldsymbol{x}$ es de mejor calidad que la mejor solución $\boldsymbol{x}_{\text{opt}}$ obtenida hasta el momento por el algoritmo, $f(\boldsymbol{x}) < f(\boldsymbol{x}_{\text{opt}})$ en caso de minimización o $f(\boldsymbol{x}) > f(\boldsymbol{x}_{\text{opt}})$ en caso de maximización, entonces se almacena como tal, $\boldsymbol{x}_{\text{opt}} := \boldsymbol{x}$.
7. Si no se cumple el criterio de terminación del algoritmo, se repiten de nuevo los pasos 4, 5 y 6. En caso contrario, el algoritmo termina, siendo la solución del problema la mejor solución actual obtenida por el algoritmo, $\boldsymbol{x}_{\text{opt}}$.

Las investigaciones de Hoos y Stützle [HOOS05] certifican que la búsqueda local iterativa es una técnica sencilla y eficaz para evitar que el proceso se detenga prematuramente en óptimos locales de baja calidad.

Como se observa, la búsqueda local iterativa es una técnica modular compuesta de cuatro bloques importantes: generación de una solución inicial, búsqueda local, perturbación, y aceptación.

- La generación de la solución inicial se puede realizar de manera aleatoria o mediante una heurística. En un estudio de Stützle [STUT03] se demuestra que para ejecuciones largas la elección de la solución inicial no es crítica.

- La búsqueda local permite encontrar un óptimo local dentro de una región reducida del espacio de soluciones. Su eficacia está ligada a la estructura de vecindad empleada, la cual depende directamente del problema tratado, si bien pueden existir diferentes estructuras de vecindad para cada problema.

- La perturbación suele implementarse mediante una función de transformación aleatoria de soluciones que también utilice una estructura de vecindad, lo que evita perturbaciones excesivamente grandes, pero diferente y de mayor magnitud que la estructura de vecindad del procedimiento de búsqueda local de bajo nivel, con lo que se evita que la búsqueda quede atrapada en el entorno del óptimo local actual.

- El criterio de aceptación de soluciones permite mejorar notablemente la solución final por lo que este módulo es estratégico en el desarrollo del algoritmo. Existen diferentes estrategias. Se puede dirigir la búsqueda hacia la máxima intensificación, eligiendo una nueva solución sólo si ésta es mejor que la anterior (búsqueda local iterativa determinista), o se puede favorecer la diversificación aceptando soluciones peores con una cierta frecuencia (búsqueda local iterativa estocástica, recocido simulado iterativo). Existe también la posibilidad de utilizar estructuras de memoria

para evitar el análisis de soluciones ya tratadas (búsqueda tabú iterativa).

Finalmente, el criterio de terminación del algoritmo suele ser un límite máximo de iteraciones sin mejora en la calidad de las nuevas soluciones respecto a la mejor actual. Este límite es arbitrario, pero puede elegirse, por ejemplo, de modo que implique (con probabilidad cercana a 1) que en la búsqueda local aleatoria de alto nivel se hayan explorado todos los óptimos locales vecinos a la solución en curso respecto a la estructura de vecindad usada en la función transformativa de perturbación.

2.3. ALGORITMOS TRANSFORMATIVOS DE BÚSQUEDA SIMPLE II: RECOCIDO SIMULADO

El recocido simulado (*Simulated Annealing*, SA) es uno de los más significativos algoritmos generales de optimización combinatoria existentes. Pertenece al grupo de los algoritmos transformativos de búsqueda simple, es decir, algoritmos que toman como base procedimental la búsqueda local. Kirkpatrick, Gelatt y Vecchi [KIRK83], e independientemente Cerny [CERN85], introdujeron el concepto del recocido simulado en el campo de la optimización combinatoria. Está basado en la fuerte analogía que puede establecerse entre la evolución de un sistema físico sometido a una fuente de calor y la resolución de grandes problemas de optimización combinatoria. Este algoritmo puede verse como una variante de la meta-heurística de *búsqueda local*, en la que se incorpora un criterio estocástico de aceptación de soluciones de peor calidad respecto a la solución en curso, con el fin de evitar que el algoritmo quede atrapado prematuramente en óptimos locales. Sin embargo, el criterio estocástico no es ciego o arbitrario, sino que se inspira en la física de los sistemas sometidos a una fuente de calor, concretamente en el proceso de enfriamiento de los metales (recocido) empleado en la industria metalúrgica como tratamiento para conseguir ciertas propiedades en el metal.

En 1953, Metropolis, Rosenbluth, Rosenbluth, A. Teller y E. Teller [METR53] desarrollaron un sencillo algoritmo basado en la mecánica estadística para simular la evolución de un sólido sometido a una fuente de calor. Este algoritmo utiliza técnicas de Monte Carlo, y genera una secuencia de transiciones de estado del sólido para una temperatura dada, hasta alcanzar el estado de equilibrio térmico. La regla de transición de estados descrita se denomina *criterio de Metropolis*, y el algoritmo en conjunto se llama *algoritmo de Metropolis*. Si se aplica el algoritmo de Metropolis en la simulación del proceso de enfriamiento del sólido, éste alcanzará el estado de mínima energía, bajo el supuesto de que para cada valor de la temperatura se consigue el equilibrio térmico.

El recocido simulado es una variante estocástica de la búsqueda local basada en el algoritmo de Metropolis para simulación de sistemas físicos sometidos a una fuente de calor. La analogía que se establece entre un problema de optimización combinatoria y un sistema físico de múltiples partículas, viene dada por las siguientes equivalencias: las soluciones del problema de optimización combinatoria son equivalentes a los estados del sistema físico, el coste de una solución es equivalente a la energía de un estado y la temperatura del sistema físico se considera en el problema de optimización combinatoria como un parámetro de control positivo. El algoritmo del recocido simulado puede verse como una iteración del algoritmo de Metropolis para valores decrecientes del parámetro de control de la temperatura.

La implementación práctica del algoritmo del recocido simulado puede realizarse generando una secuencia finita de valores decrecientes de la temperatura, y un número finito de transiciones de estado para cada valor de la temperatura. Para ello debe especificarse un *programa de enfriamiento*, o conjunto de parámetros cuyo objetivo es controlar la evolución del algoritmo. El programa de enfriamiento propuesto por Kirkpatrick, Gelatt y Vecchi [KIRK83] está compuesto por tres parámetros: valor inicial de la temperatura, función de decremento de la temperatura y número de transiciones de estado. Los dos primeros especifican la secuencia finita de valores de la temperatura, y el

último indica el número finito de transiciones de estado para cada valor de la misma.

De esta forma, dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), la estructura básica del algoritmo del recocido simulado es la siguiente:

1. Se establece como solución inicial en curso una solución candidata viable elegida al azar, $\boldsymbol{x} := \boldsymbol{x}_0$.
2. Se almacena la solución en curso ***x*** como mejor solución obtenida hasta el momento por el algoritmo, $\boldsymbol{x}_{\text{opt}} := \boldsymbol{x}$.
3. Se inicializa la temperatura o parámetro de control, $T := T_0$, así como el contador de transiciones de estado para cada valor de temperatura, $t := 0$.
4. Mediante una función generatriz aleatoria de soluciones vecinas viables *g*, se obtiene a partir de ***x*** una nueva solución candidata ***y***, $\boldsymbol{y} := g(\boldsymbol{x})$.
5. Si la nueva solución generada ***y*** es de mejor calidad que la actual ***x***, $f(\boldsymbol{y}) < f(\boldsymbol{x})$ en caso de minimización o $f(\boldsymbol{y}) > f(\boldsymbol{x})$ en caso de maximización, entonces la nueva solución ***y*** pasa a ser la solución en curso, $\boldsymbol{x} := \boldsymbol{y}$. En este caso, si la nueva solución en curso ***x*** es de mejor calidad que la mejor solución $\boldsymbol{x}_{\text{opt}}$ obtenida hasta el momento por el algoritmo, $f(\boldsymbol{x}) < f(\boldsymbol{x}_{\text{opt}})$ en caso de minimización o $f(\boldsymbol{x}) > f(\boldsymbol{x}_{\text{opt}})$ en caso de maximización, entonces se almacena como tal, $\boldsymbol{x}_{\text{opt}} := \boldsymbol{x}$. Por el contrario, si la nueva solución generada ***y*** es de peor calidad que la actual ***x***, y se cumple:

   $$e^{\frac{-|f(\boldsymbol{x})-f(\boldsymbol{y})|}{T}} > random[0,1[$$

   se acepta igualmente ***y*** como solución en curso, $\boldsymbol{x} := \boldsymbol{y}$.
6. Se incrementa el contador de transiciones de estado, $t := t + 1$.
7. Si el número *t* de transiciones de estado (pasos 4-6) efectuadas para el valor actual de la temperatura es igual a un límite *L*, se decrementa el valor de la temperatura, $T := T - \Delta T$, y se reinicializa el contador de transiciones de estado para cada temperatura, $t := 0$.

8. Si no se cumple el criterio de terminación del algoritmo, se repiten de nuevo los pasos 4-7. En caso contrario, el algoritmo termina, siendo la solución del problema la mejor solución actual obtenida por el algoritmo, $\boldsymbol{x}_{opt}$.

El criterio de terminación consiste en llegar a un límite máximo de transiciones de estado o iteraciones sin cambios en la solución en curso $\boldsymbol{x}$. En general, este límite máximo de iteraciones debe implicar (con probabilidad cercana a 1) que en el recorrido aleatorio se han explorado todas las soluciones vecinas a la solución en curso.

La característica más importante del algoritmo del recocido simulado es que, además de aceptar transiciones que suponen una mejora en el coste de la solución, también permite aceptar transiciones que suponen una pérdida de calidad de la solución. Inicialmente, para valores grandes de la temperatura, se aceptan grandes pérdidas de calidad de la solución. Al decrecer la temperatura, sólo se aceptan pérdidas de calidad más pequeñas. Finalmente, cuando la temperatura se aproxima a cero, no se acepta ninguna pérdida de calidad. Es decir, en este punto el recocido simulado se comporta igual que la búsqueda local. Esta característica significa que el algoritmo del recocido simulado puede escapar de los mínimos locales, aceptando transiciones intermedias hacia soluciones de mayor coste en beneficio de una mejor solución final.

Respecto a la función de decremento de la temperatura, puede decirse que ha sido objeto de estudio en numerosos trabajos de investigación [LAAR87], [DOWS01], [LUKE95], [LOCA00]. Se puede encontrar información detallada sobre esta importante meta-heurística en libros como Reeves [REEV93], Díaz, Glover, Ghaziri y González [DIAZ96], Michalewicz y Fogel [MICH00]. También es posible considerar hibridaciones del SA con otras meta-heurísticas, como la búsqueda local iterativa (recocido simulado iterativo), o los algoritmos evolutivos (algoritmos evolutivos con recocido simulado).

En el capítulo tres se analiza específicamente el recocido simulado debido a que es el algoritmo en el que se centra esta investigación.

2.4. ALGORITMOS TRANSFORMATIVOS DE BÚSQUEDA SIMPLE III: BÚSQUEDA TABÚ

La búsqueda tabú (*Tabu search*, TS) es una meta-heurística transformativa trayectorial que utiliza un procedimiento de búsqueda local dotado de un criterio no monótono de aceptación de soluciones, junto con una memoria, o lista tabú, que le permite detectar soluciones ya visitadas del espacio, de forma que se pueden saltar los óptimos locales y se evitan los ciclos en el proceso de búsqueda. Fue propuesta en 1986, por Fred Glover [GLOV86]. Los principios fundamentales de la búsqueda tabú fueron desarrollados en una serie de artículos de finales de los años 80 y principios de los 90, y que posteriormente se publicaron en el libro "*Tabu Search*" en 1997 [GLOV97].

El primer elemento que caracteriza a la búsqueda tabú es una memoria que guía la búsqueda hacia zonas del espacio de soluciones que aún no han sido exploradas. En la versión más básica del algoritmo esta memoria se implementa mediante una lista, denominada *lista tabú*, en la que se van introduciendo las soluciones candidatas que ya se han visitado en el proceso de búsqueda, las cuales pasan a ser *soluciones tabú* y no pueden repetirse de nuevo mientras sigan en la lista. Normalmente, las soluciones tabú no permanecen indefinidamente en esta memoria, sino que salen tras un tiempo en ella existiendo diferentes criterios de salida, denominados en general *criterios de aspiración*, y que obviamente están ligados a la complejidad de la memoria tabú implementada. En el caso básico, la lista tabú se implementa mediante una lista FIFO de longitud *n*, de modo que una vez llena, cada vez que se almacena en la lista una solución recién visitada, se debe eliminar de ella la solución registrada en la iteración de exploración *n* veces anterior a la actual. Por tanto, tras un número de iteraciones igual a la longitud de la lista tabú, las soluciones pierden su condición "prohibida" y pueden volver a ser visitadas.

El segundo elemento característico de la búsqueda tabú es el criterio de aceptación no monótono de nuevas soluciones. Así, en cada iteración del algoritmo se recorre la vecindad de la solución en curso, normalmente de forma

determinista, descartando siempre aquellas soluciones vecinas que se hallen en la lista tabú, y se selecciona como nueva solución en curso la mejor solución de la vecindad. Obviamente, esta solución no tiene por qué ser de mejor calidad que la solución en curso anterior, de ahí el carácter no monótono del criterio de aceptación, cuya principal ventaja es que permite escapar de óptimos locales. Además, al considerarse en este paso sólo las soluciones vecinas no prohibidas, se evitan los ciclos sin fin en el proceso de búsqueda.

El algoritmo básico de búsqueda tabú comienza tomando como solución inicial en curso una solución candidata viable del problema, y una lista tabú vacía. A partir de entonces se explora completamente la vecindad viable de la solución en curso, y se obtiene la mejor solución de esa vecindad, excluidas la propia solución en curso y las que pudieran coincidir con las registradas en la lista tabú hasta ese momento. Entonces, la solución en curso se introduce en la cabecera de la lista tabú (si la lista tabú excede su tamaño máximo se saca la solución en la cola de la lista), y la mejor solución encontrada en la vecindad de la solución actual pasa a ser la nueva solución en curso, repitiéndose de nuevo la exploración. Obsérvese que la nueva solución encontrada en la vecindad no tiene por qué ser mejor que la solución a la que sustituye, lo que permite escapar de los óptimos locales. El algoritmo termina cuando se ha realizado un número prefijado de iteraciones de exploración sin que se produzcan cambios respecto a la mejor solución global encontrada durante la búsqueda.

De esta forma, dado un problema de optimización combinatoria (X,S,f,R), la estructura básica del algoritmo de búsqueda tabú es la siguiente:

1. Se establece como solución inicial en curso una solución candidata viable elegida al azar, $\boldsymbol{x} := \boldsymbol{x}_0$.
2. Se almacena la solución en curso $\boldsymbol{x}$ como mejor solución obtenida hasta el momento por el algoritmo, $\boldsymbol{x}_{\text{opt}} := \boldsymbol{x}$.
3. Se inicializa la lista tabú, $L := \varnothing$.
4. Mediante una función generatriz determinista de soluciones vecinas viables g, se obtiene a partir de $\boldsymbol{x}$ la primera solución vecina $\boldsymbol{y}$ que no está en la lista tabú $\boldsymbol{y} := g(\boldsymbol{x}) \notin L$.

5. Se almacena la solución candidata vecina ***y*** como mejor solución vecina obtenida hasta el momento, $\boldsymbol{y}_{\text{opt}} := \boldsymbol{y}$.
6. Mediante la función generatriz determinista de soluciones vecinas viables *g*, se obtiene a partir de ***x*** la siguiente solución vecina ***y*** que no está en la lista tabú $\boldsymbol{y} := g(\boldsymbol{x}) \notin L$.
7. Si la solución vecina generada ***y*** es de mejor calidad que la actual mejor de la vecindad $\boldsymbol{y}_{\text{opt}}$, $f(\boldsymbol{y}) < f(\boldsymbol{y}_{\text{opt}})$ en caso de minimización o $f(\boldsymbol{y}) > f(\boldsymbol{y}_{\text{opt}})$ en caso de maximización, entonces la nueva solución vecina ***y*** pasa a ser la mejor solución de la vecindad, $\boldsymbol{y}_{\text{opt}} := \boldsymbol{y}$.
8. Si existen soluciones viables en la vecindad de la solución actual aún no procesadas, se reiteran los pasos 6 y 7. En caso contrario:
   - La solución en curso ***x*** se introduce en la cabecera de lista tabú. Si la longitud de la lista tabú excede el tamaño máximo se extrae la solución de la cola de la lista.
   - La mejor solución vecina generada $\boldsymbol{y}_{\text{opt}}$ pasa a ser la solución actual, $\boldsymbol{x} := \boldsymbol{y}_{\text{opt}}$.
   - Si la nueva solución en curso ***x*** es de mejor calidad que la mejor solución $\boldsymbol{x}_{\text{opt}}$ obtenida hasta el momento por el algoritmo, $f(\boldsymbol{x}) < f(\boldsymbol{x}_{\text{opt}})$ en caso de minimización o $f(\boldsymbol{x}) > f(\boldsymbol{x}_{\text{opt}})$ en caso de maximización, entonces se almacena como tal, $\boldsymbol{x}_{\text{opt}} := \boldsymbol{x}$.
9. Si no se cumple el criterio de terminación del algoritmo, se repiten de nuevo los pasos 4-8. En caso contrario, el algoritmo termina, siendo la solución del problema la mejor solución actual obtenida por el algoritmo, $\boldsymbol{x}_{\text{opt}}$.

Un aspecto importante a considerar en este algoritmo es el tamaño de la lista tabú. De no limitar el tamaño de la lista tabú la complejidad tanto espacial como temporal del algoritmo sería inasumible. Sin embargo, para evitar los ciclos sin fin la longitud de la lista tabú debe ser suficientemente grande como para que, cuando una solución visitada sea eliminada de la lista, la búsqueda esté en un

punto del espacio de soluciones tan alejado que no pueda alcanzarse de nuevo esa solución. En general, la longitud de la lista tabú determina el tamaño de la parte del espacio de soluciones en que se concentra la búsqueda: una lista tabú pequeña limita la búsqueda a zonas pequeñas del espacio de soluciones, mientras que una lista tabú grande permite buscar en zonas más grandes del espacio.

Existen muchas variantes del algoritmo básico de búsqueda tabú que se ha expuesto. Las principales son:

- Búsqueda tabú con memoria basada en atributos. No se almacenan soluciones completas visitadas, sino atributos (partes de una solución, diferencias entre soluciones) que pueden identificar no a una sola solución sino a un conjunto de soluciones. El atributo representativo depende generalmente del problema específico a resolver, y una buena elección del mismo puede hacer la búsqueda mucho más eficiente.

- Búsqueda tabú con memorias a corto y largo plazo. Utiliza dos listas tabú, una estándar o memoria a corto plazo, y una segunda o memoria a largo plazo basada en la frecuencia en que se produce un mismo tipo de soluciones.

- Búsqueda tabú con elección probabilística. Útil para decidir entre soluciones vecinas candidatas con igual o similar valor objetivo.

Esta meta-heurística ha generado buenos resultados para resolver problemas de optimización combinatoria con un alto grado de dificultad, especialmente aquellos que surgen en aplicaciones del mundo real [GLOV06]. En la literatura aparecen numerosas variantes y modificaciones de la búsqueda tabú. Se pueden encontrar en Glover y Laguna [GLOV93], [GLOV97] y Osman y Kelly [OSMA96].

2.5. ALGORITMOS TRANSFORMATIVOS POBLACIONALES I: ALGORITMOS EVOLUTIVOS

2.5.1. Algoritmos evolutivos: Conceptos básicos

Los algoritmos evolutivos (*evolutionary algorithms*, EA) son meta-heurísticas transformativas poblacionales de optimización inspiradas en los procesos de evolución genética de las especies. Tuvieron su origen en el estudio de los autómatas celulares dirigido por John Holland en la Universidad de Michigan en 1973 [HOLL73]. Como los algoritmos trayectoriales de búsqueda local, recocido simulado, y búsqueda tabú, operan sobre soluciones completas que se van transformando, pero en lugar de considerar una única solución en cada iteración del algoritmo, utilizan un conjunto de soluciones o población. En este sentido, pueden interpretarse como algoritmos de búsqueda múltiple paralela en el espacio de soluciones del problema.

Los algoritmos evolutivos se han convertido en métodos flexibles y robustos para la resolución de problemas complejos de optimización en disciplinas como son: la Industria, Matemática, Economía, Telecomunicaciones, Bioinformática, etc. [BAKE87], [GOLD89], [DAVI91], [SENT06], [AMIR07], [PEZZ08]. Además, los algoritmos evolutivos han sido utilizados en otras áreas diferentes a la optimización combinatoria, tales como la simulación de sistemas.

Siguiendo la terminología de la teoría de la evolución, las entidades que representan las soluciones del problema se denominan individuos, y el conjunto de estos, población. Un algoritmo evolutivo procede de forma iterativa sobre los individuos pertenecientes a la población, modificándolos mediante operadores de variación, como el cruce, la mutación y la selección, con el fin de obtener sucesivas generaciones de individuos mejorados. El criterio de finalización de estos algoritmos consiste habitualmente en alcanzar un número global máximo de generaciones (iteraciones), o bien un número máximo de generaciones sin mejoras.

El operador de cruce o recombinación consiste en una mezcla de la información de dos individuos, denominados *padres* o *progenitores*, para obtener uno o más individuos nuevos, llamados *hijos*. Por el contrario, el operador de mutación consiste en una alteración parcial aleatoria de un individuo para obtener otro individuo similar. Por último, el operador de selección tiene una doble función, ya que por una parte permite determinar los individuos que se reproducen en cada generación (mediante cruce y/o mutación), y por otra permite decidir qué individuos sobreviven y conforman la siguiente generación. En general, la selección de individuos reproductores y de individuos supervivientes se realiza tomando como base una función de adaptación al entorno o *fitness*, directamente relacionada con el objetivo del problema de optimización, de manera que los individuos que representan mejores soluciones al problema tienen más posibilidades de ser reproductores y de sobrevivir, de manera que la población progresivamente va mejorando.

2.5.2. Algoritmos genéticos

Los algoritmos genéticos (*genetic algorithms*, GA) constituyen el tipo principal de algoritmos evolutivos, y se basan en los procesos de selección natural y evolución de las especies. Su origen está en los estudios realizados a principios de los años 70 por J. H. Holland [HOLL73], [HOLL75], y posteriormente por D. E. Goldberg [GOLD89]. Estos algoritmos hacen evolucionar una población de individuos sometiéndola a acciones semejantes a las que actúan en la evolución biológica – *cruces y mutaciones genéticas* –, así como también a una selección de acuerdo con algún criterio en función del cual se decide cuáles son los individuos más adaptados, y que por ello tienen más posibilidades de reproducirse y/o de sobrevivir.

Los algoritmos genéticos comienzan estableciendo un conjunto de *m* soluciones iniciales, normalmente elegidas al azar, denominado población inicial de individuos. Cada individuo-solución está definido genéticamente por *n* genes (que representarían a las *n* variables del problema de optimización), agrupados en uno o más cromosomas (grupos homogéneos de variables). En

numerosos problemas de optimización ocurre que el conjunto de genes se agrupa en un solo cromosoma, por lo que en estos casos suelen identificarse los conceptos de individuo y de cromosoma. La función objetivo del problema en caso de maximización, o su inversa en caso de minimización, se considera una función de adaptación al entorno de los individuos.

A continuación, se van produciendo cíclicamente nuevas generaciones de individuos. En cada generación, se realizan las siguientes acciones:

- *Selección de reproductores*: Se seleccionan los individuos que van a reproducirse en la generación actual. La selección puede ser completamente aleatoria o bien realizarse de modo que los individuos con mejor adaptación al entorno tengan mayor probabilidad de ser seleccionados. Suele aceptarse la repetición en la selección de reproductores, es decir, que un mismo individuo figure varias veces en el conjunto de reproductores, con lo que puede reproducirse más de una vez en la misma generación.

- *Cruce o recombinación*: El cruce opera sobre pares de individuos reproductores para formar dos individuos descendientes cuyos genes resultan de la combinación de los genes de ambos padres.

- *Mutación*: La mutación modifica algunos genes de un individuo para formar otro similar al original.

- *Selección de supervivientes*: Se determina qué individuos entre los preexistentes y los nuevos deben sobrevivir. Para ello se emplean múltiples criterios de selección evolutiva, tanto deterministas como estocásticos. Los criterios más comunes son el *elitista*, que consiste en seleccionar los mejores individuos de la población, y el mixto *elitista-aleatorio*, que consiste en seleccionar la mitad de los supervivientes entre los mejores individuos y la otra mitad al azar entre los restantes individuos.

En el proceso evolutivo, el tamaño de la población debe mantenerse estable tras cada generación. Por lo general, el número de individuos reproductores seleccionados (contando las posibles repeticiones) suele tomarse igual al tamaño *m* de la población inicial. De esta forma, tras los procesos de cruce y mutación se tendrán un total de 2*m* individuos en la población, de manera que en la selección de supervivientes se descartarán *m* individuos, hasta recuperar el tamaño original de la población.

El cruce y la mutación pueden aplicarse de forma conjunta y dependiente generando primero nuevos individuos por cruce y después produciendo en ellos la mutación de algunos genes, o también de forma conjunta pero independiente generando algunos nuevos individuos por cruce de una parte de los reproductores y otros por mutación de los restantes reproductores. Incluso pueden emplearse ratios de cruce y mutación que especifican la proporción en que se debe aplicar cada operador. De hecho, en los algoritmos genéticos estándar el operador de cruce se utiliza en cada generación en una proporción mucho mayor que el operador de mutación. Normalmente el porcentaje de cruces es superior al 75%. También es posible considerar una variante de los algoritmos genéticos en los que sólo se emplea el operador de mutación, denominados específicamente algoritmos evolutivos. El tamaño de la población y el operador de cruce se asocian a la diversificación o capacidad de exploración del algoritmo, mientras que la mutación se asocia a la intensificación o capacidad de explotación.

En los algoritmos genéticos la implementación de los operadores de cruce y mutación debe ser específica del problema tratado, con el objetivo de generar nuevos individuos que sean viables respecto a las restricciones, y pueda asegurarse la obtención de una solución final factible del problema.

La calidad de la solución obtenida y el tiempo empleado dependen del número de individuos que se tomen para la población, de la eficiencia de los operadores de cruce y/o mutación empleados, y del número total de generaciones que se produzcan.

Hoy en día se han publicado numerosos trabajos de investigación sobre esta meta-heurística [BUCK92], [MICH92], [MITC96], [REEE03]. Estos algoritmos pueden aplicarse con éxito en numerosos problemas de optimización combinatoria, incluso combinados con heurísticas específicas [DAVI85], [JOHN95], [RAHM99], [CALG99], [AHN02], [BAKE03], [DIAZ03], [ALVA03], [MATT04], [WATA05], [CHAN05], [CHAN06], [MOON08].

2.5.3. Algoritmos meméticos

Los algoritmos meméticos (*Memetic Algorithms*, MA) son también algoritmos transformativos poblacionales de optimización aproximada, que pueden considerarse una variante híbrida de los algoritmos genéticos combinados con la técnica trayectorial de búsqueda local. Fueron definidos originalmente por P. Moscato [MOSC89], [MOSC99], [MOSC03], [MOSC04], y deben su nombre al concepto de "meme" o "gen de la memoria", introducido por R. Dawkins en su teoría del Darwinismo Universal [DAWK76], en la que intenta explicar la evolución cultural del hombre como un proceso análogo al de la evolución genética de las especies propuesta por Charles Darwin.

El procedimiento general de un algoritmo memético es muy parecido al de un algoritmo genético. Se comienza estableciendo un conjunto de soluciones iniciales aleatorias o población inicial de agentes o individuos, y a continuación se va produciendo cíclicamente un proceso de selección de reproductores, generación de nuevos individuos mediante cruce y mutación, y selección de supervivientes, hasta que se cumpla el criterio de fin establecido. Adicionalmente, cada vez que se generan nuevos individuos-solución se produce para cada uno de ellos un proceso llamado *aprendizaje cultural*, que tiene como fin mejorar su valor de adaptación. En su versión más sencilla el aprendizaje cultural transforma un organismo $\boldsymbol{x}$ en un organismo mejorado $\boldsymbol{x}'$ aplicando un algoritmo de búsqueda local en el que $\boldsymbol{x}$ es la solución inicial y $\boldsymbol{x}'$ es la correspondiente solución final que optimiza localmente la función de adaptación al entorno.

En el artículo de C. Cotta *"Una Visión General de los Algoritmos Meméticos"* [COTT07] aparece una descripción detallada de estos algoritmos. Existe una gran variedad de aplicaciones de los algoritmos meméticos en el campo de la optimización combinatoria, en donde estos se han obtenido buenos resultados ante problemas de optimización combinatoria. Destacamos algunos casos: problemas de particionado en grafos [MERZ00], partición de números [BERR99], empaquetado [REEV96], coloreado de grafos [COST95], problemas de asignación generalizados [CHUB97], asignación cuadrática [CARR92], [MERZ99], problema del viajante comercial [HOLS99], [MOSC92], programación de producción [QIAN08].

2.5.4. Algoritmos genéticos paralelos

Los algoritmos genéticos paralelos (*Paralell Genetic Algorithms*, PGA) surgen con el objetivo inicial de utilizar sistemas multiprocesador para reducir el tiempo de ejecución que requieren algunos problemas de elevada complejidad en su resolución mediante algoritmos genéticos tradicionales implementados de forma secuencial. Conceptualmente, la paralelización de los algoritmos genéticos consiste en separar la población de individuos en subpoblaciones que evolucionan de manera fundamentalmente independiente, pero que se comunican entre sí de alguna forma.

En la literatura se encuentran diferentes maneras de paralelizar un algoritmo genético. En general, los PGA se clasifican en tres categorías: PGA de grano fino [MAND89], modelos panmícticos [REEV93], y PGA basados en islas [WHIT99]. Pueden encontrarse algunos estudios generales sobre PGA en [BIAN93], [CHIP96], [ALBA99].

Los algoritmos genéticos paralelos de grano fino poseen una población estructurada espacialmente, y son apropiados para sistemas con múltiples procesadores trabajando en paralelo. Los procesos de selección y cruce se limitan a pequeños "barrios" (subpoblaciones) entre los que existe un solapamiento que permite la interacción.

En los modelos panmícticos, al igual que en los algoritmos genéticos convencionales, existe tan solo una población, de modo que los procesos de selección y cruce se realiza considerando la población completa, pero la evaluación de la función de adaptación de los individuos se divide entre varios procesadores.

Finalmente, los algoritmos basados en islas, también llamados multi-hogar, distribuidos, o de grano grueso, son los PGA que gozan de mayor popularidad en la comunidad científica. Poseen múltiples subpoblaciones cuya evolución se realiza fundamentalmente por separado, pero que intercambian individuos de manera ocasional mediante procesos denominados *migraciones*. Un artículo que presenta un completo y extenso estudio acerca de los PGA basados en islas, en el que pueden encontrarse varios enfoques de migración y comunicación entre diferentes subpoblaciones, es [CANT98]. Aparte de la posibilidad de su implementación paralela real, se ha podido comprobar que los algoritmos genéticos paralelos basados en islas se comportan de forma distinta a los tradicionales aun en simulaciones secuenciales en las que se comparan GA uni-poblacionales con PGA multi-poblacionales con igual número global de individuos, y que, en general, en estos casos los PGA basados en islas también mejoran la búsqueda [OSAB13].

2.6. ALGORITMOS TRANSFORMATIVOS POBLACIONALES II: OPTIMIZACIÓN POR INTELIGENCIA COLECTIVA

2.6.1. Fundamentos de la optimización por inteligencia colectiva

Dentro del mundo de las meta-heurísticas poblacionales se encuentran los algoritmos basados en inteligencia colectiva (*Swarm Intelligence*, SI). Se trata de un novedoso paradigma de inteligencia distribuida para la resolución de problemas de optimización, que toma su inspiración en ejemplos biológicos de

comportamiento colectivo resultado de interacciones entre los individuos de un determinado entorno, generalmente de carácter descentralizado y autoorganizativo.

Según Bonabeau, Dorigo y Théraulaz (1999) [BONA99], la expresión swarm intelligence fue usada por primera vez por Gerardo Beni, Suzanne Hackwood y Jing Wang en 1989, en el contexto de sistemas robóticos celulares y fue extendido para incluir el diseño de algoritmos inspirados en el comportamiento general de las colonias de insectos y otros animales sociales. Así, se incluyen en esta categoría a las hormigas (colonias), abejas (enjambres/colmenas), y ciertos pájaros (bandadas) y peces (bancos).

En los últimos años, muchos algoritmos basados en los comportamientos colectivos de estos animales se han aplicado con éxito a problemas de optimización de diferentes campos. Debido a esto, se describen a continuación los dos principales algoritmos de este tipo: La optimización por enjambre de partículas (*Particle Swarm Optimization*, PSO), que es un algoritmo poblacional inspirado en el comportamiento social de las bandadas de pájaros y los bancos de peces; y la colonia de abejas artificiales (*Artificial Bee Colony*, ABC), que es un algoritmo poblacional basado en el comportamiento de las abejas recolectoras de miel. Estos algoritmos son de gran importancia por la aplicabilidad que están demostrando y la cantidad de publicaciones que vienen generando en los últimos años.

2.6.2. Optimización por enjambre de partículas

La optimización por enjambre de partículas (*Particle Swarm Optimization*, PSO) es un algoritmo transformativo poblacional desarrollado por los investigadores Kennedy y Eberhart [KENN95] y posteriormente Shi [SHIY98], que está inspirado en el comportamiento de ciertos colectivos de animales sociales como son las bandadas de pájaros, los bancos de peces o los enjambres de abejas.

El algoritmo de optimización por enjambre de partículas comienza estableciendo un conjunto de *p* soluciones iniciales aleatorias denominado enjambre de partículas. Cada partícula-solución *k* se representa por un vector $\boldsymbol{x}_k$ de *n* coordenadas, una coordenada por cada variable del problema de optimización. Este vector indica la posición de la partícula en el espacio de búsqueda. La función objetivo del problema de optimización se considera una función de fuerza o adaptación al entorno de las partículas, por lo que si el objetivo del problema es de minimización deberá usarse como función de adaptación su inversa. A partir de aquí se van produciendo cíclicamente iteraciones en las que el algoritmo modifica la posición de cada partícula utilizando un vector velocidad $\boldsymbol{v}_k$ asociado a la partícula, según la ecuación:

$$\boldsymbol{x}_{k,t+1} = \boldsymbol{x}_{kt} + \boldsymbol{v}_{k,t+1}$$

El vector velocidad que modifica la posición de la partícula se recalcula en cada iteración del algoritmo en función de dos elementos: la tendencia de la partícula a volver a la mejor posición visitada por ella misma en su trayectoria por el espacio (comportamiento individual), y la tendencia de la partícula a dirigirse a la mejor posición encontrada por el enjambre en conjunto (interacción social). Se requieren, por tanto, dos elementos de memoria: una memoria de mejor posición individual para cada partícula *k* del enjambre, denotada $\boldsymbol{p}_k$, y una memoria de mejor posición global para el enjambre completo, denotada $\boldsymbol{p}_g$. El primer componente del vector de velocidad o tendencia individual de la partícula se calcula como la diferencia entre la mejor posición individual visitada y la posición actual de la partícula, $\boldsymbol{p}_k - \boldsymbol{p}_{kt}$, mientras que el componente de tendencia social se calcula como la diferencia entre la mejor posición global encontrada por el enjambre y la posición actual de la partícula, $\boldsymbol{p}_g - \boldsymbol{x}_{kt}$. Por tanto, la ecuación completa que determina el vector velocidad de las partículas en cada iteración del algoritmo es:

$$\boldsymbol{v}_{k,t+1} = w \cdot \boldsymbol{v}_{kt} + c_1 \cdot (\boldsymbol{p}_k - \boldsymbol{x}_{kt}) + c_2 \cdot (\boldsymbol{p}_g - \boldsymbol{x}_{kt})$$

donde, w ($0 \le w \le 1$) es un parámetro de inercia de la partícula, y c_1 ($0 \le c_1 \le 1$) y c_2 ($0 \le c_2 \le 1$) son parámetros que representan factores de aceleración de los componentes responsables del movimiento de la partícula. Estos parámetros

pueden ser constantes durante todo el procedimiento o elegirse al azar en el intervalo [0,1] en cada iteración. El algoritmo termina cuando se ha realizado un número prefijado *M* de iteraciones, presentándose como solución del problema la correspondiente a $\boldsymbol{p}_g$ o mejor posición encontrada por el enjambre de partículas en su desplazamiento por el espacio de búsqueda. Uno de los requisitos importantes para que el algoritmo sea viable es que el cálculo de la posición $\boldsymbol{x}_k$ de cada partícula en cada iteración debe tener en cuenta las restricciones del problema, de forma que las nuevas posiciones de las partículas correspondan a soluciones factibles.

La calidad de la solución obtenida y el tiempo empleado dependen del número de partículas (soluciones en curso) que se tomen para el enjambre y del número de iteraciones que se desee realizar. Por ello, la elección de los parámetros es muy importante, ya que determinan el comportamiento y la eficiencia del algoritmo de optimización. Esto ha sido objeto de abundantes investigaciones [SHIY98], [EBER00], [CARL01], [VAND01], [CLER02], [TREL03], [BRAT08], [ROCC09]. Un amplio estudio de las aplicaciones de optimización por enjambre de partículas se puede encontrar en [POLI07] y [POLI08]. También ha sido utilizado recientemente en la resolución de problemas de asignación de rutas a vehículos [MARI13], [BELM13].

En los últimos años se han desarrollado algunas variantes del algoritmo PSO, tales como el algoritmo de la luciérnaga (*Firefly Algorithm*, FA) [YANG09a], inspirado en el mecanismo de luminiscencia de las luciérnagas con el que se atraen unas a otras, la búsqueda del cuco (*Cuckoo Search*, CS) [YANG09b], basada en el comportamiento parasitario de algunas especies de cucos en combinación con la conducta de vuelo de ciertos pájaros y moscas de la fruta, y el algoritmo del murciélago (*Bat Algorithm*, BA) [YANG10], basado en el comportamiento ecolocalizador de los micromurciélagos.

Asimismo, con el objetivo de superar los inconvenientes del PSO convencional (rápida convergencia hacia óptimos locales, difícil equilibrio entre intensificación y diversificación, y la llamada “maldición de la dimensionalidad”) se han realizado diversas paralelizaciones del algoritmo PSO originándose la optimización por múltiples enjambres de partículas (*Parallel Particle Swarm*

Optimization, PPSO). Esta meta-heurística consiste esencialmente en ejecutar en paralelo múltiples versiones del PSO convencional, ya sea con la misma o diferente configuración de parámetros, incorporando además un protocolo de migraciones de partículas entre enjambres más o menos complejo. Algunos trabajos sobre esta meta-heurística pueden encontrarse en [CHANJ05], [NIU07], [XU08].

2.6.3. Optimización por colonia de abejas artificiales

La optimización por colonia de abejas artificiales (*Artificial Bee Colony*, ABC), propuesta por Karaboga en 2005 [KARA05], [KARA07], es un algoritmo transformativo poblacional basado en el comportamiento de recolección de polen de las abejas melíferas.

El modelo biológico en el que se inspira esta técnica es el de una colmena en la que cohabitan tres tipos diferentes de abejas: abejas obreras, abejas espectadoras y abejas exploradoras. Estas abejas tienen como objetivo la búsqueda y explotación de fuentes de alimento (polen). Con respecto a la recolección del polen, cada tipo de abeja tiene un comportamiento diferente:

- Abejas obreras: Cada abeja obrera está asociada a una fuente de alimento y comparte información sobre la misma, como la ubicación o la concentración de polen.
- Abejas espectadoras: Son abejas desocupadas que esperan en la colmena a que una abeja obrera les dé información sobre su fuente de alimento.
- Abejas exploradoras: Son las abejas que se encargan de buscar nuevas fuentes de alimento.

La forma de compartir información de las abejas melíferas es por medio de una danza en la que la abeja obrera gira sobre sí misma y hace vibrar su abdomen repetidamente siempre en una misma posición, de modo que la posición indica a las abejas espectadoras la dirección de la fuente de alimento y la duración de

la vibración les indica la concentración de polen de la fuente. Así, cuanto más rentable resulte la fuente, más larga será la danza y por lo tanto, mayor será la probabilidad de que una abeja espectadora la observe y elija explotar dicha fuente de alimento convirtiéndose en ese momento en obrera. Cuando se agota una fuente de alimento, la correspondiente abeja obrera tiene que elegir entre volverse abeja espectadora y esperar a recibir información de una abeja obrera sobre su fuente de alimento para explotarla, o bien convertirse en abeja exploradora y salir a buscar nuevas fuentes de alimento.

El algoritmo de colonia de abejas artificiales se basa en el comportamiento anterior para encontrar soluciones a problemas de optimización. En el algoritmo ABC las fuentes de alimento representan las soluciones del problema, el fitness de la solución es representado mediante la cantidad de polen de la fuente, y las abejas son agentes que operan sobre las fuentes de alimento emulando los tres tipos de comportamiento descritos. Así, las abejas obreras son agentes que explotan la fuentes de alimento realizando una búsqueda en la vecindad de la solución representada por la fuente, las abejas espectadoras son agentes que esperan a recibir información de fuentes de alimento prometedoras, y las abejas exploradoras buscan nuevas fuentes de alimento en zonas del espacio de soluciones sin explorar.

El algoritmo ABC comienza generando un conjunto de soluciones iniciales y evaluándolas. Cada una de estas soluciones iniciales es una fuente de alimento. El siguiente paso es explotar las distintas fuentes de alimento, es decir, realizar modificaciones a la solución inicial obteniendo nuevas soluciones en su vecindad. En función de lo prometedores que sean los resultados, se dedicará mayor o menor tiempo a explotar la fuente de alimento tal y como se comportan las abejas realizando su danza en la colmena y provocando que otras abejas (las abejas espectadoras) exploten también esa fuente de alimento. En el modelo propuesto por D. Karaboga, las nuevas soluciones se obtienen combinando las soluciones candidatas. Tras un cierto número de ciclos sin mejorar la solución, se considera que la fuente de alimento se ha agotado, y se abandona guardando la mejor solución encontrada. Finalmente, las abejas obreras asociadas a esa fuente de alimento, se convierten en abejas

desocupadas (espectadoras o exploradoras). El algoritmo continúa buscando nuevas fuentes de alimento mediante abejas exploradoras y explotándolas mediante abejas obreras durante un número de iteraciones definidas.

Desde los primeros estudios de Karaboga en 2005, numerosos investigadores han estado estudiando el algoritmo de optimización por colonia de abejas artificiales y sus aplicaciones a problemas del mundo real, tal como Hadidi [HADI10], Zhang [ZHAN11], y Garitselov [GARI12]. En el artículo de Karaboga [KARA12] puede encontrarse un estudio detallado acerca del ABC.

Por último, en este caso también se han realizado diversas paralelizaciones del algoritmo ABC originándose la optimización por múltiples colonias de abejas artificiales (*Parallel Artificial Bee Colony*, PABC). La primera versión del algoritmo PABC fue propuesta en el año 2009 con el objetivo de mejorar la eficiencia de los ABC simples [TSAI09]. Como puede suponerse, la base de los algoritmos PABC es la ejecución en paralelo de varias instancias del ABC convencional. Además, se incorpora la migración de abejas entre colonias o el intercambio de información entre las diferentes colonias que forman el sistema. Algunas implementaciones interesantes del PABC son [EL-AB10], [BANH10], [PARP10].

2.7. ALGORITMOS TRANSFORMATIVOS POBLACIONALES III: OPTIMIZACIÓN POR COMBINACIÓN DE SOLUCIONES

2.7.1. Fundamentos de la optimización por combinación de soluciones

Las meta-heurísticas poblacionales de tipo evolutivo y de inteligencia de enjambre operan a lo largo de todo el proceso de búsqueda sobre un conjunto de soluciones en lugar de sobre una única solución, por lo que suelen presentar tiempos de computación sensiblemente más altos que las meta-heurísticas de búsqueda simple. A esta situación, hay que añadirle que la

convergencia de la población necesita de un gran número de iteraciones. Con el fin de encontrar procedimientos que obtengan soluciones de calidad más rápidamente se han diseñado otros algoritmos poblacionales basados en la combinación de soluciones.

La optimización por combinación de soluciones puede fundamentarse en el principio de que *la información sobre la calidad de un conjunto de soluciones puede utilizarse para su combinación y mejora* [ALBA03]. Además, este tipo de algoritmos utiliza la inclusión de procesos paralelos y descentralizados en los que la población se divide con algún criterio de entorno y se hace evolucionar de forma separada a zonas distintas dentro de la tradicional población única [ALBA02].

A continuación se describen brevemente dos algoritmos de este tipo, que han demostrado su efectividad en los últimos años: la búsqueda dispersa y el reencadenamiento de trayectorias.

2.7.2. Búsqueda dispersa

La búsqueda dispersa (*Scatter Search*, SS) es una meta-heurística basada en la combinación de soluciones que tiene su origen en los años setenta y se ha aplicado con éxito a la resolución de numerosos problemas de optimización, recibiendo una gran atención por parte de la comunidad científica. En un artículo de Glover de 1998 [GLOVE98] se analiza de manera específica la búsqueda dispersa y se recogen ideas expuestas en trabajos anteriores. Esta publicación tuvo un gran impacto para la difusión de esta meta-heurística, y numerosos investigadores comenzaron a aplicar la búsqueda dispersa para la resolución de problemas de optimización obteniendo buenos resultados.

Aunque se trata de un método similar a los algoritmos evolutivos, presenta diferencias importantes con respecto a estos. La búsqueda dispersa trabaja sobre un pequeño conjunto de soluciones denominado conjunto de referencia. Básicamente consiste en combinar las soluciones que forman el conjunto de referencia para obtener nuevas soluciones que mejoren a las que las

originaron. Posteriormente, a estas soluciones se les aplica un procedimiento de mejora de búsqueda local. Este procedimiento está basado en la premisa de que cuando se combinan soluciones y se aplica un método de mejora se logran mejores resultados que cuando se aplica el método de mejora en las soluciones originales sin combinarlas previamente.

Esquemáticamente, el algoritmo de búsqueda dispersa se desarrolla de la forma siguiente:

- El algoritmo comienza con la generación de un conjunto inicial de soluciones diversas.
- A continuación se realiza una mejora de dichas soluciones mediante un procedimiento de búsqueda local.
- Posteriormente se construye el conjunto de referencia (*Refset*) con las mejores soluciones siguiendo determinados criterios de calidad y diversidad.
- Una vez obtenido el conjunto de referencia inicial, se realiza un proceso cíclico en el que se van generando subconjuntos con las soluciones del conjunto de referencia, se combinan las soluciones de cada subconjunto para obtener soluciones distintas a las de partida, y se mejoran con el procedimiento de búsqueda local, para finalmente actualizar el conjunto de referencia seleccionando las soluciones que sean mejores por calidad o por diversidad. El proceso se repite hasta que se dé un ciclo en el que no se obtengan soluciones que puedan ser incorporadas al conjunto de referencia.

Aunque los conceptos y principios fundamentales de la búsqueda dispersa, fueron propuestos al comienzo de la década de los sesenta, la estrategia se puede considerar relativamente moderna y en constante desarrollo. Durante los últimos años se han desarrollado numerosos trabajos aplicando la búsqueda dispersa a la resolución tanto de conocidos problemas de optimización como de problemas del mundo real. Estos trabajos han posibilitado nuevos campos

de estudio. Laguna y Martí realizan una revisión exhaustiva de la búsqueda dispersa en [LAGU02], [LAGU03], [MART03].

2.7.3. Reencadenamiento de trayectorias

El reencadenamiento de trayectorias (*Path Relinking*, PR) [GLOVE98], [LAGU00], [LAGU03], es una meta-heurística poblacional, variante de la búsqueda dispersa, basada en una estrategia de intensificación que explora las trayectorias que interconectan las soluciones del conjunto de referencia. Fundamentalmente, esta meta-heurística se caracteriza por que explora un camino entre cada par de soluciones seleccionadas en lugar de generar una sola solución por combinación de las dos seleccionadas. Por tanto, trata de generar nuevas soluciones por combinación explorando exhaustivamente las trayectorias que conectan las soluciones del conjunto de referencia.

Como la búsqueda dispersa, el reencadenamiento de trayectorias comienza con la generación de un conjunto inicial de soluciones diversas a las que se les aplica una mejora por búsqueda local, para construir el conjunto de referencia con las mejores soluciones obtenidas. A continuación, se divide el conjunto de referencia en dos partes: un subconjunto de soluciones de iniciación (initiating solutions) y un subconjunto de soluciones guía (guiding solutions). Por cada par de soluciones formado por una solución de iniciación y una solución guía el algoritmo explora un camino entre ambas en el espacio de soluciones. Este proceso se lleva a cabo introduciendo atributos de la solución guía en la solución inicial. El recorrido se define tomando cada vez el atributo de la solución guía que lo hace más cercano a ella. Posteriormente, el conjunto de referencia se actualiza con una o más de las soluciones obtenidas en el recorrido anterior.

Como se observa, la diferencia entre el reencadenamiento de trayectorias y la búsqueda dispersa reside en la forma de actualizar el conjunto de referencia. En el caso de los algoritmos de reencadenamiento de trayectorias, solo se actualiza el conjunto de referencia si una nueva solución encontrada en el camino trazado entre un par de soluciones, mejora la peor solución del

conjunto. En ese caso, se reemplaza la peor solución por la solución encontrada, manteniendo siempre un conjunto de referencia de tamaño fijo.

2.8. ALGORITMOS CONSTRUCTIVOS I: ALGORITMOS VORACES

2.8.1. Algoritmos voraces simples

Los algoritmos voraces simples (*Simple Greedy Algorithms*, SGA), también llamados golosos o miopes, son algoritmos constructivos de búsqueda simple que forman paso a paso la solución del problema seleccionando secuencialmente los valores de las variables que producen la mejor solución parcial. Así, un algoritmo voraz comienza estableciendo una solución parcial inicial denominada *semilla*, formada por un subconjunto de variables con valores asignados. Esta solución parcial inicial puede incluso ser vacía, es decir, sin variables asignadas. En cada iteración se debe asignar valor a una de las variables no asignadas del problema. Los valores asignables a la variable en curso que verifican las restricciones del problema respecto a las variables ya asignadas se llaman *valores candidatos*. Una vez que se ha obtenido la lista de valores candidatos para la variable actual se selecciona aquel que produzca una mejora más elevada en la calidad de la solución parcial para esa iteración. Cuando un candidato es rechazado, lo es definitivamente, y si es seleccionado, permanecerá en la solución durante todo el proceso. El algoritmo termina cuando todas las variables del problema han sido asignadas.

Los métodos voraces reducen dinámicamente el espacio de soluciones de forma que cada vez que un candidato es aceptado se elimina toda la región del espacio correspondiente al resto de los candidatos. El principal inconveniente es que en cada iteración se selecciona sólo el mejor candidato para la variable actual lo que puede tener como consecuencia construir en posteriores iteraciones peores soluciones globales. Esto se conoce como *miopía* del

proceso e implica que el algoritmo voraz finaliza en soluciones que son óptimos locales del problema, es decir, no es un algoritmo de gran calidad en cuanto a optimización [BRAS97]. De hecho, la calidad de los algoritmos voraces está en relación con las características del problema que se intenta resolver: un algoritmo voraz puede arrojar muy buenos resultados para determinadas instancias del problema y no para otras.

Una descripción detallada de este tipo de algoritmos puede encontrarse en [CORM01]. Trabajos como los realizados por: Chand y Scheneeberger [CHAN98], Gutin, Yeo y Zverovich [GUTI02], Campello y Maculan [CAMP92], y Tupia [TUPI01] son algunos ejemplos de aplicaciones de algoritmos voraces para resolver problemas de optimización combinatoria que aparecen en la literatura.

2.8.2. GRASP

El procedimiento de búsqueda voraz aleatoria adaptativa o algoritmo GRASP (acrónimo de *Greedy Randomized Adaptive Search Procedure*) es una meta-heurística constructiva desarrollada originalmente por T. Feo y M. Resende [FEO89], [FEO95].

El algoritmo GRASP es un procedimiento multi-arranque en el que cada arranque se corresponde con una iteración formada por dos fases claramente diferenciadas:

- En primer lugar, una *fase constructiva* que consiste en obtener una solución completa mediante un algoritmo voraz modificado. En esta fase, el algoritmo voraz comienza estableciendo una solución parcial inicial denominada *semilla*, formada por un subconjunto de variables con valores asignados. En cada paso se debe asignar valor a una de las variables no asignadas del problema entre un conjunto de *candidatos* (valores asignables a la variable en curso que verifican las restricciones del problema respecto a las variables ya asignadas). Una vez que se ha obtenido la lista básica de candidatos se debe seleccionar uno de ellos

para asignarlo a la variable actual, lo que se realiza con un procedimiento estocástico diferente a la selección determinista del algoritmo voraz estándar. En primer lugar, se ordena la lista básica de candidatos en función del coste que generan en la solución parcial. A continuación, se construye una nueva lista que agrupa a los mejores candidatos para ser seleccionados o lista RCL (*Restricted Candidate List*) aplicando como criterio de restricción un umbral de coste μ dado por la fórmula:

$$\mu = c_{\min} + \alpha \cdot (c_{\max} - c_{\min})$$

donde $c_{\min}$ y $c_{\max}$ son los costes mínimo y máximo asociados a los candidatos de la lista base, y α $(0 \leq \alpha \leq 1)$ es un parámetro que determina el tamaño de la lista RCL y que puede variar adaptándose dinámicamente en cada arranque del algoritmo. Todos los candidatos de la lista base con un coste inferior o igual al umbral μ se incorporan a la lista RCL. Una vez que se ha obtenido la lista RCL, el candidato final que se debe asignar a la variable en curso se selecciona al azar de esta lista. Como se observa, en el contexto del algoritmo GRASP no se selecciona el mejor candidato, sino uno de los mejores, lo que permite al algoritmo en su funcionamiento global superar los óptimos locales constructivos. Este proceso se repite con la siguiente variable por asignar, y la fase termina cuando todas las variables del problema han sido asignadas.

- En segundo lugar, se realiza una *fase de mejora*, en donde se optimiza la solución obtenida en la fase de construcción mediante un algoritmo de búsqueda local estándar. La búsqueda local de mejora comienza tomando como solución inicial en curso la solución construida por el algoritmo voraz. A continuación se van generando secuencialmente nuevas soluciones vecinas a la solución en curso y para cada nueva solución generada se comprueba si mejora la calidad frente a la solución en curso, de manera que en caso afirmativo la nueva solución mejor pasa a ser la solución en curso. La mejora termina cuando se ha explorado totalmente (o con probabilidad cercana a 1) el conjunto de soluciones vecinas a la actual sin que se produzcan mejoras cualitativas.

El ciclo se repite hasta satisfacer un criterio de parada, que consiste normalmente en realizar un número fijo de iteraciones. La mejor solución obtenida entre todas las iteraciones es la que presenta el algoritmo como resultado final.

Como se ha mencionado, el parámetro α determina el tamaño de la lista RCL, de forma que si $\alpha = 0$, en la lista RCL sólo estará el mejor candidato (selección miope pura), mientras que si $\alpha = 1$, en la lista RCL estarán todos los candidatos base (selección aleatoria pura). Existen varias estrategias para elegir el parámetro α, que normalmente varía dinámicamente en cada arranque del algoritmo:

- Seleccionar su valor al azar de acuerdo con una distribución de probabilidad uniforme.
- Ajustar su valor de acuerdo con la calidad de las soluciones recientes obtenidas.
- Comenzar con un valor $\alpha = 1$ (selección aleatoria pura) y reducirlo linealmente en cada arranque del algoritmo hasta llegar al valor $\alpha = 0$ (selección miope pura) en el último arranque.

Algunas aplicaciones de la meta-heurística GRASP a diferentes problemas de optimización combinatoria se recogen en [FEST02], [RESE03].

2.9. ALGORITMOS CONSTRUCTIVOS II: OPTIMIZACIÓN POR COLONIA DE HORMIGAS

Con unos principios similares a los de la optimización por inteligencia colectiva (de hecho, mucho autores clasifican esta meta-heurística en ese grupo), pero bajo una filosofía constructiva, Dorigo, Maniezzo y Colorni [DORI96] plantean un algoritmo poblacional constructivo de optimización llamado *optimización por colonia de hormigas* (*Ant Colony Optimization*, ACO). Se inspira en el experimento con hormigas reales de Goss en el que se muestra la capacidad

que tienen las hormigas para encontrar el camino más corto de ida y vuelta entre el hormiguero y las fuentes de alimento. En un principio las hormigas se mueven de forma aleatoria, pero según pasa el tiempo la mayoría de las hormigas selecciona el camino más corto. Este comportamiento se produce gracias al mecanismo de transmisión de información propio de las hormigas. En efecto, cuando una hormiga encuentra comida, regresa al hormiguero dejando un rastro de una sustancia química denominada *feromona*. Así, cuando una hormiga intenta encontrar alimento sigue la ruta que contiene el rastro más fuerte de feromonas dejado por otras hormigas, y si encuentra la comida, de vuelta al hormiguero provoca a su vez un aumento del nivel de feromonas en ese camino, reforzándolo. Por tanto, la probabilidad que tiene una hormiga de escoger un camino crece en función del número de hormigas que lo hayan escogido previamente. En función del tiempo transcurrido, el rastro de feromonas empieza a evaporarse y, por lo tanto, se reduce la atracción de las feromonas. La densidad de feromonas es más intensa en recorridos cortos puesto que su uso es más frecuente. La evaporación de feromonas tiene la ventaja de eludir que los caminos escogidos por la primera hormiga resulten demasiado atractivos para el resto evitando que el espacio de búsqueda sea limitado.

La optimización basada en colonias de hormigas es un procedimiento multi-arranque en el que se usan estrategias análogas al comportamiento de las colonias de hormigas en busca de alimento en la vida real. En cada arranque o iteración global, se parte de una población o colonia inicial de hormigas artificiales. Cada hormiga artificial es un agente computacional simple que debe construir una solución del problema mediante la utilización de feromonas artificiales. Las hormigas artificiales comienzan el proceso saliendo de la colonia, lo que equivale a partir de una solución parcial del problema o solución semilla, que incluso puede ser vacía. En cada paso de la construcción de la solución completa la hormiga artificial debe hacer un desplazamiento simple, lo que significa asignar valor a una de las variables no asignadas del problema. Obviamente, los valores asignables a la variable en curso serán aquellos que verifican las restricciones del problema respecto a las variables ya asignadas. El criterio de decisión de una hormiga para desplazarse está basado en una

función estocástica que depende de la cantidad de feromonas artificiales que haya en el camino y del objetivo de optimización. Tras cada iteración global debe actualizarse el nivel de feromona, mediante un proceso denominado *evaporación*. En [DORI97], [STUT00] se puede obtener información sobre la forma de actualizar el nivel de feromonas. La evaporación de feromona evita que el algoritmo converja demasiado rápidamente posibilitando la exploración de nuevas áreas de búsqueda. Tras actualizarse el nivel de feromonas comienza una nueva iteración global. La solución al problema de optimización es la mejor solución encontrada en todas las iteraciones ejecutadas.

Los algoritmos de colonias de hormigas tienen una naturaleza discreta, lo cual los hace muy interesantes para la resolución de problemas de optimización combinatoria. El primer algoritmo básico de hormigas desarrollado, el Ant System [DORI91], [COLO91], [COLO92], tuvo su primera aplicación en la solución del problema del viajante comercial. En estos primeros trabajos ya se proponen variaciones al concepto de feromona, derivándose tres algoritmos del Ant System: el *Ant Quantity*, el *Ant Density* y el *Ant Cycle*. Otros problemas solucionados por medio del Ant System son la versión asimétrica del TSP, el problema de la asignación cuadrática y el problema de la programación de producción discreta [DORI96].

Otros algoritmos de colonias de hormigas son: el *ANT–Q*, presentado por Gambardella y Dorigo [GARB95]; *Ant Colony System*, propuesto por Dorigo y Gambardella [DORI97]; el min–max *Ant System*, propuesto por Stützle y Hoos [STUT00]; el *Rank Based Ant System*, presentado por Bullnheimer [BULL97]; el *Best–Worst Ant System*, desarrollado por Cordón [CORD00]. Otras variaciones en el concepto de la actualización de la feromona se pueden encontrar en Meuleau y Dorigo [MEUL00].

Capítulo 3: Recocido Simulado

“Un hombre con una nueva idea es
un loco hasta que ésta triunfa”

Mark Twain

3.1. INTRODUCCIÓN AL ALGORITMO RECOCIDO SIMULADO

El recocido simulado (*Simulated Annealing*, SA) es uno de los más significativos algoritmos generales de optimización combinatoria existentes. Pertenece al grupo de los algoritmos transformativos de búsqueda simple, es decir, algoritmos que toman como base procedimental la búsqueda local. Este algoritmo puede verse como una variante de la meta-heurística de *búsqueda local*, en la que se incorpora un criterio estocástico de aceptación de soluciones de peor calidad respecto a la solución en curso, con el fin de evitar que el algoritmo quede atrapado prematuramente en óptimos locales. Sin embargo, el criterio estocástico no es ciego o arbitrario, sino que se inspira en la física de los sistemas sometidos a una fuente de calor, concretamente en el proceso de enfriamiento de los metales (recocido) empleado en la industria metalúrgica como tratamiento para conseguir ciertas propiedades en el metal.

Para proceder al estudio del recocido simulado, se presentan a continuación los fundamentos físicos y se describe detalladamente este algoritmo heurístico.

3.1.1. Fundamentos físicos del recocido simulado

En la primera mitad de la década de los ochenta, Kirkpatrick, Gelatt y Vecchi [KIRK83], e independientemente Cerny [CERN85], introdujeron el concepto del recocido simulado en el campo de la optimización combinatoria. Este concepto se basa en la fuerte analogía que puede establecerse entre la evolución de un sistema físico sometido a una fuente de calor y la resolución de grandes problemas de optimización combinatoria. En los párrafos siguientes se desarrollan – de forma intuitiva – algunos conceptos básicos que constituyen el fundamento del recocido simulado y que permiten comprender correctamente dicha analogía.

La *mecánica estadística* es una rama de la física que estudia los sistemas formados por un número tan elevado de partículas que el comportamiento del sistema no puede determinarse evaluando la contribución de cada partícula por separado. En lugar de esto, se asume que el sistema físico es compatible con una población estadística cuyos parámetros coinciden con los parámetros observables del sistema físico. De esta forma, puede determinarse, por ejemplo, la energía media del sistema a partir de la media estadística de las energías de las partículas constituyentes. Si P_i es la probabilidad de que una partícula *i* tenga la energía E_i, entonces la energía media del sistema es:

$$\overline{E} = \sum_i P_i E_i$$

Una clase muy importante de este tipo de sistemas físicos es la formada por los que están en contacto con una *fuente de calor*. Las fuentes de calor se caracterizan porque toda interacción con el sistema en cuestión da lugar únicamente a cambios infinitesimales en las propiedades de la fuente, mientras que el sistema puede sufrir cambios de importancia hasta que se alcance el equilibrio térmico, el cual viene determinado por la denominada *distribución de Boltzmann* [TODA83]. Esta distribución proporciona la probabilidad de que el sistema físico se halle en un estado *i* con una energía E_i a la temperatura absoluta *T*, y viene dada por la expresión:

$$P(i,T) = \frac{e^{\frac{-E_i}{k_B T}}}{Z(T)}$$

donde:

- k_B es una constante física conocida como *constante de Boltzmann*,
- el factor exponencial $e^{\frac{-E_i}{k_B T}}$ se denomina *factor de Boltzmann*, y
- *Z(T)* es la *función de partición*, definida como:

$$Z(T) = \sum_j e^{\frac{-E_j}{k_B T}}$$

  donde el sumatorio se extiende sobre el conjunto de estados del sistema.

En la física de la materia condensada, el recocido (annealing) es un proceso térmico destinado a obtener estados de baja energía en un sólido sometido a una fuente de calor. Este proceso consiste fundamentalmente en los dos pasos siguientes:

- Incrementar la temperatura de la fuente de calor hasta el punto de fusión del sólido, momento en el que las partículas del mismo se disponen aleatoriamente.
- Decrementar paulatinamente la temperatura de la fuente a fin de que las partículas del sólido se organicen en una estructura cristalina que representa el estado de mínima energía del sólido.

El estado de mínima energía se obtiene sólo si la fase de enfriamiento se realiza suficientemente despacio. En caso contrario, las partículas del sólido se disponen en estructura amorfa resultando un estado estable de alta energía en lugar del deseado de mínima energía. En otras palabras, se alcanza un mínimo local, pero no global.

El proceso del recocido puede modelizarse con éxito mediante métodos de simulación por computador. Ya en 1953, N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller y E. Teller [METR53] desarrollaron un sencillo algoritmo basado en la mecánica estadística para simular la evolución de un sólido sometido a una fuente de calor. Este algoritmo utiliza técnicas de Monte Carlo, y genera una secuencia de *L* transiciones de estado del sólido para una temperatura dada *T*, hasta alcanzar el estado de equilibrio térmico. Cada una de las transiciones de estado se realiza de la siguiente forma. Dado un estado actual *i* del sólido con energía E_i, se aplica un mecanismo de perturbación que produce un pequeño cambio – por ejemplo, el desplazamiento de una partícula –, en el estado actual *i*, para generar un nuevo estado *j* del sólido. Si la energía E_j del nuevo estado es menor que la del estado actual E_i, entonces se acepta el estado *j* como el estado actual del sólido. En caso contrario, el estado *j* se acepta como estado actual con una probabilidad dada por la expresión:

$$e^{\frac{E_i - E_j}{k_B T}}$$

donde;

- k_B es la constante de Boltzmann, y
- *T* es la temperatura absoluta de la fuente de calor.

La regla de transición de estados descrita se denomina *criterio de Metropolis*, y el algoritmo en conjunto se llama *algoritmo de Metropolis*. Si se aplica el algoritmo de Metropolis en la simulación del proceso de enfriamiento del sólido, éste alcanzará el estado de mínima energía, bajo el supuesto de que para cada valor de la temperatura se consigue el equilibrio térmico.

3.1.2. Algoritmo del recocido simulado

El recocido simulado es una variante estocástica de la búsqueda local basada en el algoritmo de Metropolis para simulación de sistemas físicos sometidos a una fuente de calor. La analogía que se establece entre un problema de

optimización combinatoria y un sistema físico de múltiples partículas, viene dada por las siguientes equivalencias:

- Las soluciones del problema de optimización combinatoria son equivalentes a los estados del sistema físico.
- El coste de una solución es equivalente a la energía de un estado.
- La temperatura del sistema físico se considera en el problema de optimización combinatoria como un parámetro de control $T > 0$.

El algoritmo del recocido simulado puede verse ahora como una iteración del algoritmo de Metropolis para valores decrecientes del parámetro de control *T*.

Algoritmo (Recocido simulado). Dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), la estructura básica del algoritmo es la siguiente:

1. Se establece como solución inicial en curso una solución candidata viable elegida al azar, $\boldsymbol{x} := \boldsymbol{x}_0$.
2. Se almacena la solución en curso ***x*** como mejor solución obtenida hasta el momento por el algoritmo, $\boldsymbol{x}_{\text{opt}} := \boldsymbol{x}$.
3. Se inicializa la temperatura o parámetro de control, $T := T_0$, así como el contador de transiciones de estado para cada valor de temperatura, $t := 0$.
4. Mediante una función generatriz aleatoria de soluciones vecinas viables *g*, se obtiene a partir de ***x*** una nueva solución candidata ***y***, $\boldsymbol{y} := g(\boldsymbol{x})$.
5. Si la nueva solución generada ***y*** es de mejor calidad que la actual ***x***, $f(\boldsymbol{y}) < f(\boldsymbol{x})$ en caso de minimización o $f(\boldsymbol{y}) > f(\boldsymbol{x})$ en caso de maximización, entonces la nueva solución ***y*** pasa a ser la solución en curso, $\boldsymbol{x} := \boldsymbol{y}$. En este caso, si la nueva solución en curso ***x*** es de mejor calidad que la mejor solución $\boldsymbol{x}_{\text{opt}}$ obtenida hasta el momento por el algoritmo, $f(\boldsymbol{x}) < f(\boldsymbol{x}_{\text{opt}})$ en caso de minimización o $f(\boldsymbol{x}) > f(\boldsymbol{x}_{\text{opt}})$ en caso de maximización, entonces se almacena como tal, $\boldsymbol{x}_{\text{opt}} := \boldsymbol{x}$. Por el

contrario, si la nueva solución generada ***y*** es de peor calidad que la actual ***x***, y se cumple:

$$e^{\frac{-|f(\boldsymbol{x})-f(\boldsymbol{y})|}{T}} > random[0,1[$$

se acepta igualmente ***y*** como solución en curso, $\boldsymbol{x} := \boldsymbol{y}$.

6. Se incrementa el contador de transiciones de estado, $t := t + 1$.
7. Si el número *t* de transiciones de estado (pasos 4-6) efectuadas para el valor actual de la temperatura es igual a un límite *L*, se decrementa el valor de la temperatura, $T := T - \Delta T$, y se reinicializa el contador de transiciones de estado para cada temperatura, $t := 0$.
8. Si no se cumple el criterio de terminación del algoritmo, se repiten de nuevo los pasos 4-7. En caso contrario, el algoritmo termina, siendo la solución del problema la mejor solución actual obtenida por el algoritmo, $\boldsymbol{x}_{opt}$.

El criterio de terminación consiste en llegar a un límite máximo de transiciones de estado o iteraciones sin cambios en la solución en curso ***x***. En general, este límite máximo de iteraciones debe implicar (con probabilidad cercana a 1) que en el recorrido aleatorio se han explorado todas las soluciones vecinas a la solución en curso.

La característica más importante del algoritmo del recocido simulado es que, además de aceptar transiciones que suponen una mejora en el coste de la solución, también permite aceptar, hasta un cierto límite, transiciones que suponen una pérdida de calidad de la solución. La probabilidad de aceptar transiciones con pérdida de calidad se implementa comparando el valor de la expresión:

$$e^{\frac{-|f(\boldsymbol{x})-f(\boldsymbol{y})|}{T}}$$

con un número aleatorio generado a partir de una distribución uniforme sobre el intervalo [0,1[. Inicialmente, para valores grandes de la temperatura *T*, se aceptan grandes pérdidas de calidad de la solución. Al decrecer *T*, sólo se

aceptan pérdidas de calidad más pequeñas. Finalmente, cuando la temperatura se aproxima a cero, no se acepta ninguna pérdida de calidad. Es decir, en este punto el recocido simulado se comporta igual que la búsqueda local. Esta característica significa que el algoritmo del recocido simulado puede escapar de los mínimos locales, aceptando transiciones intermedias hacia soluciones de mayor coste en beneficio de una mejor solución final.

En virtud de la analogía con el algoritmo de Metropolis (simulación del proceso físico del recocido), el algoritmo del recocido simulado debe converger asintóticamente hacia el conjunto de soluciones óptimas globales del problema. No obstante, para garantizar esta convergencia asintótica es necesaria la existencia de una única distribución estacionaria para el algoritmo del recocido simulado, equivalente a la existencia de la distribución de Boltzmann de equilibrio térmico en mecánica estadística. E. Aarts y J. Korst proporcionan una demostración completa de esta conjetura en su libro *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing* [AART89], utilizando para ello un formalismo matemático del algoritmo del recocido simulado basado en la teoría de las cadenas de Markov. Asimismo, se proporcionan las condiciones suficientes de convergencia asintótica del recocido simulado hacia el conjunto de soluciones óptimas globales del problema [AART89]. Igualmente, otros autores han demostrado condiciones similares de convergencia asintótica del recocido simulado [GEMA84], [MITR86], mientras que Hajek ha proporcionado condiciones necesarias y suficientes [HAJE88]. Esencialmente, la condición de convergencia al óptimo global establece que la temperatura T del sistema debe decrecer de manera logarítmica de acuerdo con la ecuación:

$$T_k = \frac{T_0}{1+\ln(1+k)}$$

donde $k = 0, 1, ..., n$ indica el ciclo de temperatura. Sin embargo, esta función de enfriamiento del sistema requiere unos tiempos de computación prohibitivos, por lo que es necesario plantear métodos de decremento de la temperatura más rápidos (programas de enfriamiento) que permitan usar el recocido simulado como algoritmo de optimización aproximada.

La implementación práctica del algoritmo del recocido simulado puede realizarse generando una secuencia finita de valores decrecientes de la temperatura *T*, y un número finito *L* de transiciones de estado para cada valor de la temperatura. Para ello debe especificarse un *programa de enfriamiento*, o conjunto de parámetros cuyo objetivo es controlar la evolución del algoritmo.

El siguiente programa de enfriamiento usado frecuentemente en la literatura fue propuesto por Kirkpatrick, Gelatt y Vecchi [KIRK83], y se compone de tres parámetros. Los dos primeros especifican la secuencia finita de valores de la temperatura, y el último indica el número finito de transiciones de estado para cada valor de la misma:

- *Valor inicial de la temperatura*, T_0. Este valor inicial de la temperatura debe ser suficientemente elevado como para que se acepte con una cierta probabilidad cercana a 1 cualquier nueva solución generada en una transición de estado.

- *Función de decremento de la temperatura*. Generalmente, se utiliza una función de decremento exponencial de la forma $T_k = T_0 \cdot \alpha^k$, donde α es una constante un poco menor que la unidad. Los valores usuales de α oscilan entre 0,8 y 0,99.

- *Número de transiciones de estado*, *L*, para cada valor de la temperatura. Intuitivamente, el número de transiciones para cada temperatura debe ser suficientemente grande como para que, si no se aceptasen cambios de solución, pudiera recorrerse con una probabilidad cercana a 1 todo el conjunto de soluciones vecinas viables a la solución en curso.

Respecto a la temperatura inicial T_0, en la práctica puede determinarse mediante el siguiente algoritmo:

Algoritmo (Cálculo de la temperatura inicial en el recocido simulado): Dado un problema de optimización combinatoria (*X*,*S*,*f*,*R*), el cálculo de la temperatura inicial T_0 en el recocido simulado se realiza mediante el siguiente procedimiento:

1. Se elige al azar una solución candidata viable, $\boldsymbol{x} := \boldsymbol{x}_0$.

2. Se inicializa la temperatura o parámetro de control T con un valor positivo ε para el cual la función de Boltzmann $e^{\frac{-|f(\boldsymbol{x})-f(\boldsymbol{y})|}{T}}$ tome valores inferiores a 0'5 (lo que supone rechazar frecuentemente las soluciones de peor calidad), $T := \varepsilon$. Por ejemplo, ε puede ser el valor absoluto de la diferencia entre el coste de dos soluciones candidatas viables elegidas al azar, de modo que:

$$e^{\frac{-|f(\boldsymbol{x})-f(\boldsymbol{y})|}{T}} = e^{-1} \cong 0'368$$

3. Se inicializa el contador de transiciones de estado para cada valor de temperatura, $t := 0$, así como el contador de transiciones de estado aceptadas que supongan deterioro de calidad, $t' := 0$.

4. Mediante una función generatriz aleatoria de soluciones vecinas viables *g*, se obtiene a partir de $\boldsymbol{x}$ una nueva solución candidata $\boldsymbol{y}$, $\boldsymbol{y} := g(\boldsymbol{x})$, repitiéndose este paso hasta que la nueva solución generada $\boldsymbol{y}$ sea de peor calidad que la actual $\boldsymbol{x}$, $f(\boldsymbol{y}) > f(\boldsymbol{x})$ en caso de minimización o $f(\boldsymbol{y}) < f(\boldsymbol{x})$ en caso de maximización.

5. Si se cumple:

$$e^{\frac{-|f(\boldsymbol{x})-f(\boldsymbol{y})|}{T}} > random[0,1[$$

   se incrementa el contador de transiciones de estado aceptadas, $t' := t' + 1$.

6. Se incrementa el contador de transiciones de estado, $t := t + 1$.

7. Si $t < 10$, se repiten los pasos 4-6.

8. Si $t' < t$, se incrementa la temperatura multiplicándola por un factor β mayor que la unidad, $T := \beta T$, siendo usualmente $1'1 \leq \beta \leq 1'5$, y se repiten los pasos 3-7. En caso contrario, el algoritmo termina, siendo la temperatura inicial apropiada para el recocido simulado igual a la temperatura *T* alcanzada en el proceso, $T_0 := T$.

En la analogía con el sistema físico, este algoritmo coincide con el calentamiento del sólido hasta el punto de fusión, en el que todas las partículas del mismo se disponen al azar.

Otro método para calcular la temperatura inicial T_0 de forma más directa – pero que requiere un conocimiento específico del problema de optimización a resolver –, consiste en despejar T_0 en la ecuación:

$$e^{\frac{\sup \Delta f}{T_0}} = P(\boldsymbol{y})$$

es decir,

$$T_0 = \frac{\sup \Delta f}{\ln(P(\boldsymbol{y}))}$$

donde $P(\boldsymbol{y})$ es una probabilidad de aceptación de soluciones peores cercana a la unidad (por ejemplo, 0.95), y sup Δf es una cota superior del incremento medio de coste entre soluciones factibles del problema.

Respecto a la función de decremento de la temperatura, puede decirse que ha sido objeto de estudio en numerosos trabajos de investigación [LAAR87], [DOWS01], [LUKE95], [LOCA00], y es el interés principal de la comparativa desarrollada a continuación.

3.2. PROGRAMAS DE ENFRIAMIENTO

El recocido simulado es una de las más importantes meta-heurísticas, siendo bien conocidas sus propiedades de convergencia hacia soluciones de alta calidad, aunque con un elevado coste computacional. Esto ha dado origen a numerosos trabajos de investigación sobre aceleración de la convergencia del algoritmo, especialmente en el tratamiento del parámetro de temperatura, mediante lo que se conoce como programas o estrategias de enfriamiento. En este apartado se describen diferentes programas de enfriamiento: cuatro de

enfriamiento monótono multiplicativo, cuatro de enfriamiento monótono aditivo, y uno de enfriamiento no monótono adaptativo [DIAZ08]. Todos ellos se componen al menos de los tres parámetros planteados en el programa de Kirkpatrick, Gelatt y Vecchi:

- Temperatura inicial T_0,
- función de decremento de la temperatura, y
- número L de transiciones de estado para cada valor de la temperatura.

La temperatura inicial y el número de transiciones de estado se calculan de la misma forma en todos los casos, siendo el modo de reducir la temperatura del sistema lo que diferencia a los distintos programas de enfriamiento.

3.2.1. Enfriamiento monótono multiplicativo

En el enfriamiento monótono multiplicativo, la temperatura T del sistema en el ciclo k se calcula multiplicando la temperatura inicial T_0 por un factor decreciente con respecto al ciclo k. Se consideran cuatro variantes:

- *Enfriamiento multiplicativo exponencial*, propuesto por Kirkpatrick, Gelatt y Vecchi [KIRK83], y utilizado como referencia en la comparación entre los distintos criterios de enfriamiento. La reducción de la temperatura se realiza multiplicando la temperatura inicial T_0 por un factor decreciente exponencial respecto al ciclo de temperatura k:

$$T_k = T_0 \cdot \alpha^k$$

  donde $0.8 \leq \alpha \leq 0.99$. La curva de decrecimiento de la temperatura se caracteriza por una pendiente negativa pronunciada en los primeros ciclos de procesamiento, pero que se va suavizando progresivamente según disminuye la temperatura (figura 3.1).

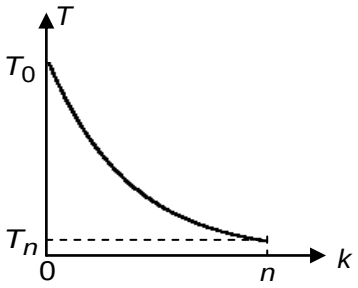

Figura 3.1: Curva de enfriamiento multiplicativo exponencial, $T_k = T_0 \cdot \alpha^k$

- *Enfriamiento multiplicativo logarítmico*, basado en la condición de convergencia asintótica del recocido simulado [AART89], pero que incorpora un factor α de aceleración del enfriamiento que posibilita su uso en la práctica. La reducción de la temperatura se realiza multiplicando la temperatura inicial T_0 por un factor decreciente inversamente proporcional al logaritmo natural del ciclo de temperatura *k*:

$$T_k = \frac{T_0}{1+\alpha \ln(1+k)}$$

donde $\alpha > 1$. La curva de decrecimiento de la temperatura se caracteriza por una pendiente negativa muy pronunciada en los primeros ciclos de procesamiento, que se va suavizando muy rápidamente según se reduce la temperatura (figura 3.2).

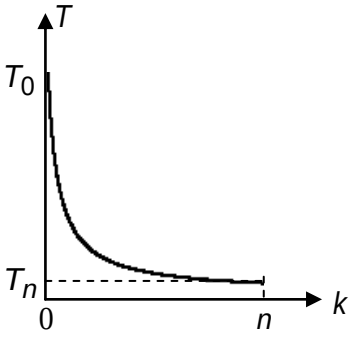

Figura 3.2: Curva de enfriamiento multiplicativo logarítmico, $T_k = \frac{T_0}{1+\alpha \ln(1+k)}$

- *Enfriamiento multiplicativo lineal*. La reducción de la temperatura se realiza multiplicando la temperatura inicial T_0 por un factor decreciente inversamente proporcional al ciclo de temperatura *k*:

$$T_k = \frac{T_0}{1+\alpha k}$$

donde $\alpha > 0$. La curva de decrecimiento de la temperatura posee una forma similar a la multiplicativa logarítmica, pero con pendiente menos pronunciada al principio y menos suave al final (figura 3.3).

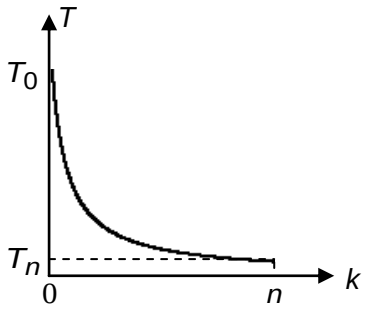

Figura 3.3: Curva de enfriamiento multiplicativo lineal, $T_k = \frac{T_0}{1+\alpha k}$

- *Enfriamiento multiplicativo cuadrático*. La reducción de la temperatura se realiza multiplicando la temperatura inicial T_0 por un factor decreciente inversamente proporcional al cuadrado del ciclo de temperatura *k*:

$$T_k = \frac{T_0}{1+\alpha k^2}$$

donde $\alpha > 0$. La curva de decrecimiento de la temperatura posee forma sigmoidal irregular con un arco convexo hacia abajo muy amplio (figura 3.4).

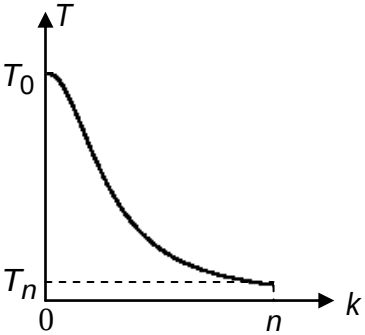

Figura 3.4: Curva de enfriamiento multiplicativo cuadrático, $T_k = \frac{T_0}{1+\alpha k^2}$

3.2.2. Enfriamiento monótono aditivo

En el enfriamiento monótono aditivo se deben considerar dos parámetros adicionales: el número *n* de ciclos de enfriamiento a realizar, y la temperatura final del sistema T_n. En este tipo de enfriamiento la temperatura *T* del sistema en el ciclo *k* se calcula sumando a la temperatura final T_n un término decreciente con respecto al ciclo *k*. Se consideran cuatro variantes, basadas en las fórmulas propuestas por B.T. Luke [LUKE05]:

- *Enfriamiento aditivo lineal*. La reducción de la temperatura se realiza sumando a la temperatura final T_n un término decreciente lineal respecto al ciclo de temperatura *k*:

$$T_k = T_n + (T_0 - T_n)\left(\frac{n-k}{n}\right)$$

  De esta manera, para $k = 0$ es $T_k = T_n + (T_0 - T_n)(1) = T_0$, y para $k = n$ es $T_k = T_n + (T_0 - T_n)(0) = T_n$. La curva de decrecimiento de la temperatura es una línea recta de pendiente negativa constante (figura 3.5).

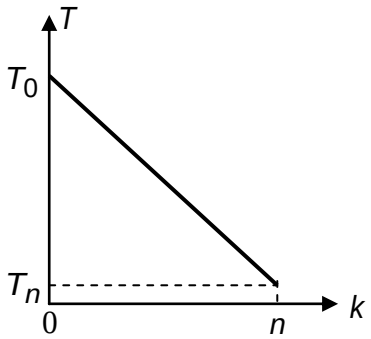

Figura 3.5: Curva de enfriamiento aditivo lineal, $T_k = T_n + (T_0 - T_n)\left(\frac{n-k}{n}\right)$

- *Enfriamiento aditivo cuadrático*. La reducción de la temperatura se realiza sumando a la temperatura final T_n un término decreciente cuadrático respecto al ciclo de temperatura *k*:

$$T_k = T_n + (T_0 - T_n)\left(\frac{n-k}{n}\right)^2$$

De esta manera, para $k = 0$ es $T_k = T_n + (T_0 - T_n)(1)^2 = T_0$, y para $k = n$ es $T_k = T_n + (T_0 - T_n)(0)^2 = T_n$. La curva de decrecimiento de la temperatura se caracteriza por una pendiente negativa pronunciada en los primeros ciclos de procesamiento, pero que se va suavizando progresivamente según se reduce la temperatura (figura 3.6).

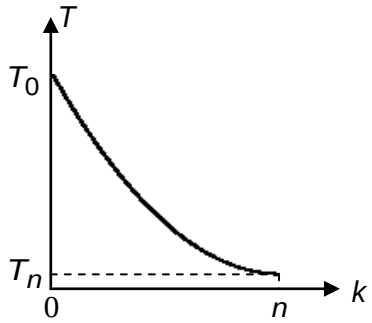

Figura 3.6: Curva de enfriamiento aditivo cuadrático,

$$T_k = T_n + (T_0 - T_n)\left(\frac{n-k}{n}\right)^2$$

- *Enfriamiento aditivo exponencial*. La reducción de la temperatura se realiza sumando a la temperatura final T_n un término decreciente inversamente proporcional a la exponencial natural respecto al ciclo de temperatura *k*:

$$T_k = T_n + (T_0 - T_n)\left(\frac{1}{1 + e^{\frac{10}{n}\left(k - \frac{1}{2}n\right)}}\right)$$

De esta manera, para $k = 0$ es $T_k = T_n + (T_0 - T_n)\left(1^-\right) \cong T_0$, y para $k = n$ es $T_k = T_n + (T_0 - T_n)\left(0^+\right) \cong T_n$. La curva de decrecimiento de la temperatura se caracteriza por su forma sigmoidal pronunciada, de pendiente muy suave al comienzo y al final del procesamiento y pronunciada en su punto central (figura 3.7).

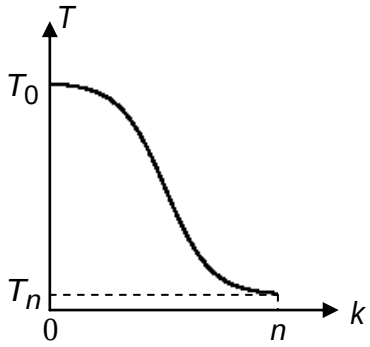

Figura 3.7: Curva de enfriamiento aditivo exponencial,

$$T_k = T_n + (T_0 - T_n)\left(\frac{1}{1+e^{\frac{10}{n}\left(k-\frac{1}{2}n\right)}}\right)$$

- *Enfriamiento aditivo trigonométrico*. La reducción de la temperatura se realiza sumando a la temperatura final T_n un término decreciente proporcional al coseno del ciclo de temperatura *k*:

$$T_k = T_n + \tfrac{1}{2}(T_0 - T_n)\left(1+\cos\left(\frac{k\pi}{n}\right)\right)$$

De esta manera, para $k = 0$ es $T_k = T_n + \frac{1}{2}(T_0 - T_n)(1+1) = T_0$, y para $k = n$ es $T_k = T_n + \frac{1}{2}(T_0 - T_n)(1-1) = T_n$. La curva de decrecimiento de la temperatura se caracteriza por su forma sigmoidal suave (figura 3.8).

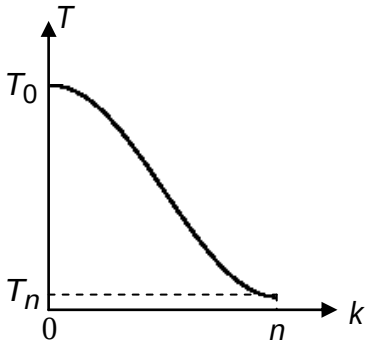

Figura 3.8: Curva de enfriamiento aditivo trigonométrico,

$$T_k = T_n + \tfrac{1}{2}(T_0 - T_n)\left(1+\cos\left(\frac{k\pi}{n}\right)\right)$$

3.2.3. Enfriamiento no monótono adaptativo

En el enfriamiento no monótono adaptativo, la temperatura *T* del sistema en cada transición de estado se calcula multiplicando el valor de temperatura T_k obtenido mediante cualquiera de los criterios precedentes por un factor adaptativo μ basado en la diferencia en valor absoluto entre el objetivo de la solución en curso, $f(\boldsymbol{x})$, y el mejor objetivo conseguido hasta el momento por el algoritmo, denotado *f**:

$$\mu = 1 + \frac{|f(\boldsymbol{x}) - f^*|}{f(\boldsymbol{x})}$$

Luego:

$$T = \mu T_k = \left(1 + \frac{|f(\boldsymbol{x}) - f^*|}{f(\boldsymbol{x})}\right) T_k$$

Como puede observarse, se verifica que $1 \leq \mu < 2$, siendo tanto mayor la temperatura – y con ello mayores los incrementos de energía permitidos – cuanto mayor sea la distancia de la solución actual a la mejor solución obtenida hasta el momento. Por tanto, se trata de un criterio de reducción de la temperatura *T* que se comporta de forma no monótona. Es decir, en cada transición de estado la temperatura *T* toma un valor que puede ser mayor o menor que el valor alcanzado por *T* en la transición anterior, si bien la tendencia de *T* es decreciente y se aproxima a cero al final del proceso de cálculo, por ser T_k decreciente y con límite cero. Este método de reducción de la temperatura es una modificación del propuesto por M. Locatelli [LOCA00], y como se ha mencionado, se puede utilizar combinándolo con cualquiera de los criterios anteriores en el cálculo de T_k. En la comparativa se ha utilizado como método complementario el enfriamiento multiplicativo exponencial estándar. En este caso, la curva de enfriamiento se caracteriza por un comportamiento oscilante aleatorio comprendido entre la curva exponencial definida por T_k y su duplo $2T_k$ (figura 3.9).

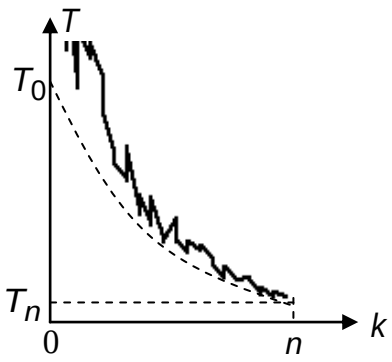

Figura 3.9: Curva de enfriamiento no monótono adaptativo $T = \left(1 + \frac{|f(\boldsymbol{x}) - f^*|}{f(\boldsymbol{x})}\right) T_k$, combinado con el enfriamientomultiplicativo exponencial, $T_k = T_0 \cdot \alpha^k$

3.3. COMPARATIVA DE PROGRAMAS DE ENFRIAMIENTO DEL RECOCIDO SIMULADO

Para completar el estudio básico del algoritmo del recocido simulado se va a realizar a continuación una prueba comparativa de los nueve programas de enfriamiento presentados en el apartado anterior. Esta comparativa toma como base un conjunto de pruebas que se han obtenido ejecutando el algoritmo del recocido simulado con cada programa de enfriamiento un total de 100 veces sobre dos problemas clásicos de optimización combinatoria utilizados como "benchmarking", un problema de planificación de rutas y otro de configuración, considerando para cada problema dos instancias reales. En la comparativa se realiza un análisis estadístico de los resultados tanto respecto a la calidad de las soluciones obtenidas como respecto a la convergencia temporal del algoritmo.

Los problemas de optimización combinatoria utilizados en las pruebas del algoritmo con las diferentes estrategias de enfriamiento son:

- *el problema del viajante comercial y*
- *el problema de la asignación cuadrática*

Se dan las pautas para la correcta elección de los parámetros de enfriamiento y se realiza el análisis de los resultados obtenidos en las pruebas.

3.3.1. Descripción de los problemas de optimización combinatoria

Problema del Viajante Comercial. El problema del viajante comercial (*Travelling Salesman Problem, TSP*) consiste en encontrar la ruta más corta de recorrido cíclico de *n* ciudades de forma que cada ciudad se visite una sola vez. Formalmente, el problema puede describirse mediante la cuádrupla *(X,S,f,R)*, donde*:*

- $X = \{x_1,..., x_n\}$, es el conjunto de variables del problema, en el que cada variable x_i representa la posición *i* de la ruta a recorrer. El dominio finito común para las *n* variables es $D = \{1,2,..., n\}$, que representa al conjunto de las *n* ciudades a visitar.

- $S = D^n$, es el espacio de soluciones del problema, de tamaño $\#S = \#D^n = n^n$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1,\ldots,x_n) \in S$.

- $f : S \to \mathbb{R}$, es la función objetivo a minimizar, o distancia de recorrido cíclico de las *n* ciudades, determinada por:

$$f(\boldsymbol{x}) = f(x_1,\ldots,x_n) = \sum_{i=1}^{n} d\left(x_i, x_{(i \bmod n)+1}\right)$$

  donde $d(x_i, x_j)$ es la distancia existente entre las ciudades x_i y x_j.

- $R = \{r_1,... ,r_n\}$, es el conjunto de restricciones:

$$r_1 : x_1 = 1$$

$$r_i : \forall j \in \{1,..., i-1\} : x_i \neq x_j \quad (i = 2,..., n)$$

  La circularidad de la ruta buscada permite fijar la primera variable x_1 con la ciudad 1 (restricción r_1), y asignar libremente las restantes ciudades 2

a n a las variables x_2 a x_n, teniendo en cuenta que no puede repetirse ninguna ciudad en la ruta (restricciones r_2 a r_n). Por tanto, las soluciones viables son permutaciones de las n ciudades con la ciudad 1 como primera ciudad a visitar. Así, dada una ruta $\boldsymbol{x}$ que verifique las restricciones, cualquier otra ruta cíclica que pase por las n ciudades en igual orden (y por tanto con igual valor objetivo asociado) será una rotación de $\boldsymbol{x}$.

Nota: La rutas correspondientes en orden inverso se consideran aquí diferentes, aunque el valor objetivo asociado sea el mismo.

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1,...,s_n) \in S$, que minimice globalmente la función objetivo f, de tal forma que se satisfagan todas las restricciones impuestas entre las variables. El espacio de soluciones viables es el conjunto de las permutaciones circulares de n ciudades: $S^* = \left\{ \boldsymbol{x} = (1, x_2,..., x_n) \in S \mid \forall i \in \{2,..., n\}, \forall j \in \{1,..., i-1\} : x_i \neq x_j \right\}$, de tamaño $\#S^* = (n-1)!$.

La función generatriz de soluciones vecinas viables g utilizada en este caso para el TSP es la correspondiente a la estructura de 2-proximidad, es decir, se trata de la función que asocia cada solución $\boldsymbol{x} \in S$ con una solución aleatoria $\boldsymbol{y}$ que difiere de $\boldsymbol{x}$ en el valor de dos variables como máximo. Dado que las soluciones candidatas viables del problema son permutaciones de las n ciudades, la función generatriz obtiene las soluciones por intercambio de los valores de dos variables elegidas al azar. De acuerdo con esto, el tamaño del conjunto de soluciones candidatas viables vecinas a una dada se calcula como el número de selecciones de 2 variables no necesariamente distintas tomadas del total de n variables del problema, lo que es igual al número de combinaciones con repetición de n objetos tomados en grupos de dos:

$$CR(n,2) = \frac{n(n+1)}{2}$$

Para realizar las pruebas de evaluación del algoritmo, se han utilizado dos instancias de la versión euclídea del TSP:

- Problema del viajante sobre un recorrido formado por las 47 capitales de provincia españolas peninsulares, siendo la distancia mínima entre dos ciudades de 47 km (Valladolid-Palencia), y la distancia máxima de 1384 km (Gerona-Cádiz).

- Problema del viajante sobre un recorrido formado por 80 ciudades europeas, siendo la distancia mínima entre dos ciudades de 6 km (Tallinn-Helsinki), y la distancia máxima de 5462 km (Tromso-Faro). Aquí las distancias utilizadas corresponden al camino más corto entre cada par de ciudades, pero con posibilidad de cruzar zonas marítimas (Canal de la Mancha, Mar Báltico) transportando el automóvil en ferry o similar. En estas situaciones no se tiene en cuenta la parte del trayecto realizada por mar.

Problema de la Asignación Cuadrática. El problema de la asignación cuadrática (*Quadratic Assignment Problem*, QAP) consiste en la ubicación de *n* plantas industriales en *n* lugares posibles, teniendo en cuenta que entre cada dos plantas existe un flujo de transporte de materiales w_{ij} con un coste directamente proporcional a la distancia entre los lugares en que se hallen las plantas, de forma que se minimice el coste total del transporte de materiales entre las plantas. Formalmente, el problema puede describirse mediante la cuádrupla *(X,S,f,R)*, donde*:*

- $X = \{x_1,...,x_n\}$, es el conjunto de variables del problema, en el que cada variable x_i representa la planta industrial número *i* a ubicar. El dominio finito común para las *n* variables es el conjunto de los *n* lugares donde se deben ubicar las plantas $D = \{1,2,...,n\}$.

- $S = D^n$, es el espacio de soluciones del problema, de tamaño $\#S = \#D^n = n^n$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1, \ldots, x_n) \in S$.
- $f : S \to \mathbb{R}$, es la función objetivo a minimizar, o coste total del transporte de materiales entre las plantas, determinada por:

$$f(\boldsymbol{x}) = f(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} d(x_i, x_j)$$

  donde $d(x_i, x_j)$ es la distancia existente entre los lugares x_i y x_j, y w_{ij}, con $i, j \in \{1, \ldots, n\}$, es la cantidad de material que debe transportarse de la planta número *i* a la planta número *j*. Intuitivamente, esta función de coste favorece la ubicación cercana de plantas entre las que exista un alto flujo de materiales.
- $R = \{r_1, \ldots, r_{n-1}\}$, es el conjunto de restricciones:

$$r_i : \forall j \in \{1, \ldots, i\} : x_{i+1} \neq x_j \quad (i = 1, \ldots, n-1)$$

  Las restricciones r_1 a r_{n-1} que relacionan las variables x_1 a x_n aseguran que cada lugar es asignado una sola vez.

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1, \ldots, s_n) \in S$, que minimice globalmente la función objetivo *f*, de tal forma que se satisfagan todas las restricciones impuestas entre las variables. El espacio de soluciones viables es el conjunto $S^* = \{\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in S \mid \forall i \in \{1, \ldots, n-1\}, \forall j \in \{1, \ldots, i\} : x_{i+1} \neq x_j\}$, formado por las permutaciones de los *n* lugares, siendo el tamaño de dicho espacio de soluciones viables $\#S^* = n!$.

La función generatriz aleatoria de soluciones vecinas viables *g* utilizada en este caso en el QAP es la correspondiente a la estructura de 2-proximidad, es decir, se trata de la función que asocia cada solución $\boldsymbol{x} \in S$ con una solución aleatoria ***y*** que difiere de ***x*** en el valor de dos variables como máximo. Dado

que las soluciones candidatas viables del problema son permutaciones de los *n* lugares sobre las *n* plantas, la función generatriz obtiene las soluciones tomando dos variables (plantas) cualesquiera e intercambiando sus valores. De acuerdo con esto, el tamaño del conjunto de soluciones candidatas viables a una dada se calcula como el número de selecciones de 2 variables no necesariamente distintas tomadas del total de *n* variables, que es igual al número de combinaciones con repetición de *n* objetos tomados en grupos de dos:

$$CR(n,2) = \frac{n(n+1)}{2}$$

Para realizar las pruebas de evaluación del algoritmo, se han utilizado dos instancias del QAP:

- Problema de la asignación cuadrática en el que se deben ubicar 47 plantas de trabajo en 47 ciudades españolas peninsulares, con cantidades mínima y máxima a transportar entre dos plantas de 0 unidades y 99 unidades respectivamente, y costes unitarios mínimo y máximo de transporte entre dos ciudades de 47 euros (Valladolid-Palencia) y 1384 euros (Gerona-Cádiz) respectivamente.

- Problema de la asignación cuadrática en el que se deben ubicar 80 plantas de trabajo en 80 ciudades europeas, con cantidades mínima y máxima a transportar entre dos plantas de 0 unidades y 99 unidades respectivamente, y costes unitarios mínimo y máximo de transporte entre dos ciudades de 6 euros (Tallinn-Helsinki) y 5462 euros (Tromso-Faro) respectivamente.

3.3.2. Elección de parámetros

Dado que la comparativa se realiza tomando como base siempre el mismo algoritmo del recocido simulado, es relativamente sencillo establecer un criterio de homogeneidad temporal entre los diferentes programas de enfriamiento comparados. En este caso, se utiliza como referencia el programa de

enfriamiento multiplicativo exponencial de Kirkpatrick, Gelatt y Vecchi con factor de decremento $\alpha = 0.95$.

En cada instancia del problema a resolver, todas las versiones del algoritmo utilizan los mismos valores de temperatura inicial T_0 y número *L* de transiciones de estado para cada valor de la temperatura. La temperatura inicial T_0 puede calcularse por los métodos descritos anteriormente. El número *L* de transiciones de estado debe asegurar, con una probabilidad η cercana a 1 (por ejemplo, $\eta = 0.95$) que, si no se aceptan cambios de solución, pueda procesarse cualquier solución vecina a la solución candidata viable en curso. Para su cálculo se procede de la siguiente forma: Si *m* es el número total de soluciones vecinas a una solución dada, e ***y*** es una solución arbitraria vecina, entonces la probabilidad de elegir a ***y*** en una transición de estado es $P(\boldsymbol{y}) = \frac{1}{m}$. De aquí, la probabilidad de que ***y*** sea elegida al menos una vez en *L* transiciones de estado es:

$$P(\boldsymbol{y},L) = P(\boldsymbol{y}) + (1-P(\boldsymbol{y}))P(\boldsymbol{y}) + (1-P(\boldsymbol{y}))^2 P(\boldsymbol{y}) + \ldots + (1-P(\boldsymbol{y}))^{L-1}P(\boldsymbol{y}) =$$

$$= \sum_{i=1}^{L}(1-P(\boldsymbol{y}))^{i-1}P(\boldsymbol{y}) = P(\boldsymbol{y})\frac{1-(1-P(\boldsymbol{y}))^L}{1-(1-P(\boldsymbol{y}))} = 1-(1-P(\boldsymbol{y}))^L = 1-\left(\frac{m-1}{m}\right)^L$$

Igualando la expresión resultante al valor de probabilidad deseado η, se obtiene el número *L* de transiciones de estado necesarias:

$$1-\left(\frac{m-1}{m}\right)^L = \eta \Rightarrow L = \frac{\ln(1-\eta)}{\ln\left(\frac{m-1}{m}\right)}$$

Para determinar los parámetros de reducción de la temperatura de cada programa de enfriamiento, se considera la temperatura final media $\overline{T}$ y su desviación típica σ correspondientes al enfriamiento multiplicativo exponencial con $\alpha = 0.95$. Evidentemente, el intervalo $\left[\overline{T}-\sigma, \overline{T}+\sigma\right]$ marca los valores más probables de la temperatura final para este programa de enfriamiento, y puede servir como base para fijar la temperatura final de los restantes programas. El objetivo es determinar los parámetros asociados a la reducción de la

temperatura de modo que se llegue a una temperatura en el intervalo $\left[\overline{T}-\sigma,\overline{T}+\sigma\right]$ en un número de ciclos similar al empleado por el enfriamiento multiplicativo exponencial. Se distinguen tres casos:

- *Enfriamiento monótono multiplicativo*. El factor de decremento α que aparece en las ecuaciones de reducción de la temperatura debe permitir llegar a la temperatura $\overline{T}+\sigma$, o temperatura más elevada, a partir de la que es muy probable la terminación del algoritmo en el mismo número *n* de ciclos que el enfriamiento multiplicativo exponencial. Sabiendo que para este enfriamiento el número *n* de ciclos es:

$$\overline{T}+\sigma = T_0 \cdot (0.95)^n \Rightarrow n = \frac{\ln\left(\frac{\overline{T}+\sigma}{T_0}\right)}{\ln(0.95)}$$

  resulta para el enfriamiento multiplicativo logarítmico:

$$\overline{T}+\sigma = \frac{T_0}{1+\alpha\ln(1+n)} \Rightarrow \alpha = \frac{\frac{T_0}{\overline{T}+\sigma}-1}{\ln(1+n)}$$

  para el enfriamiento multiplicativo lineal:

$$\overline{T}+\sigma = \frac{T_0}{1+\alpha n} \Rightarrow \alpha = \frac{\frac{T_0}{\overline{T}+\sigma}-1}{n}$$

  y para el enfriamiento multiplicativo cuadrático:

$$\overline{T}+\sigma = \frac{T_0}{1+\alpha n^2} \Rightarrow \alpha = \frac{\frac{T_0}{\overline{T}+\sigma}-1}{n^2}$$

- *Enfriamiento monótono aditivo*. El número *n* de ciclos de temperatura y la temperatura final T_n que aparecen en las ecuaciones de reducción de la temperatura de estos tipos de enfriamiento deben ser: $T_n = \overline{T}-\sigma$, o temperatura más baja en que es muy probable la terminación del

algoritmo, y *n* el correspondiente número de ciclos del enfriamiento multiplicativo exponencial para esa temperatura, es decir:

$$\bar{T} - \sigma = T_0 \cdot (0.95)^n \Rightarrow n = \frac{\ln\left(\frac{\bar{T} - \sigma}{T_0}\right)}{\ln(0.95)}$$

- *Enfriamiento no monótono adaptativo*. Dado que en este caso el enfriamiento adaptativo se combina con el enfriamiento multiplicativo exponencial, el factor de decremento α que aparece en la ecuación de reducción de la temperatura debe ser también $\alpha = 0.95$.

3.3.3. Resultados de las pruebas

El software desarrollado para soportar el algoritmo del recocido simulado con las diferentes técnicas de enfriamiento y su aplicación a los problemas del viajante y de la asignación cuadrática, se ha realizado mediante una metodología de diseño y programación orientada a objetos en el lenguaje de programación C++. Las pruebas se han ejecutado en un computador personal con unos tiempos medios que oscilan entre 1 segundo para TSP 47 y 40 segundos para QAP 80. Ahora bien, dado que entre las diferentes pruebas la implementación del algoritmo del recocido simulado es la misma y sólo cambian los programas de enfriamiento, la proporción entre el número total de iteraciones y el tiempo de ejecución es constante en todas las pruebas, aproximadamente igual a 60.000 iteraciones por segundo. Por ello, en las tablas de resultados se proporciona como información de convergencia solo la relativa al número total de iteraciones efectuadas.

La tabla 3.1 muestra los parámetros de los programas de enfriamiento utilizados en cada instancia de los problemas TSP y QAP:

	TSP 47	TSP 80	QAP 47	QAP 80
	$T_0 = 16369{,}79$ $L = 3384$	$T_0 = 6554{,}53$ $L = 9720$	$T_0 = 269432{,}451$ $L = 3384$	$T_0 = 1300980{,}28$ $L = 9720$
M. Exponencial	$\alpha = 0{,}95$	$\alpha = 0{,}95$	$\alpha = 0{,}95$	$\alpha = 0{,}95$
M. Logarítmico	$\alpha = 350$	$\alpha = 2100$	$\alpha = 768{,}6$	$\alpha = 1030$
M. Lineal	$\alpha = 12$	$\alpha = 60{,}35$	$\alpha = 24{,}3$	$\alpha = 31{,}6$
M. Cuadrático	$\alpha = 0{,}08$	$\alpha = 0{,}333$	$\alpha = 0{,}15$	$\alpha = 0{,}19$
Monótono Aditivo	$n = 146$ $T_n = 9{,}37$	$n = 181$ $T_n = 6{,}0022$	$n = 161$ $T_n = 688{,}92$	$n = 167$ $T_n = 2464{,}9$
No monótono Adaptativo	$\alpha = 0{,}95$	$\alpha = 0{,}95$	$\alpha = 0{,}95$	$\alpha = 0{,}95$

Tabla 3.1: Parámetros de los programas de enfriamiento

Para cada instancia de los problemas se han realizado 100 ejecuciones con los nueve programas de enfriamiento, obteniéndose la media y la desviación típica tanto del objetivo logrado como del número total de iteraciones empleadas en cada caso (tablas 3.2-3.5, figuras 3.10-3.17).

		MED	DESV
E M Log	Objetivo	7439,58	343,52
	Iteraciones	142624,41	72210,02
E M Lin	Objetivo	7006	352,86
	Iteraciones	349138,45	74278,16
E M Cua	Objetivo	7030,69	382,67
	Iteraciones	447286,2	55776,80
E M Exp	Objetivo	7040,17	383,07
	Iteraciones	498561,6	28723,40
E A Lin	Objetivo	7775,08	470,25
	Iteraciones	535389,66	39308,51
E A Cua	Objetivo	7183,73	355,62
	Iteraciones	532664,19	52723,84
E A Exp	Objetivo	7072,56	293,97
	Iteraciones	502794,07	67931,19
E A Trig	Objetivo	7324,19	425,62
	Iteraciones	538035,36	60389,06
E Adapt	Objetivo	7073,42	359,56
	Iteraciones	502653,87	26854,95

Tabla 3.2: Resultados TSP 47

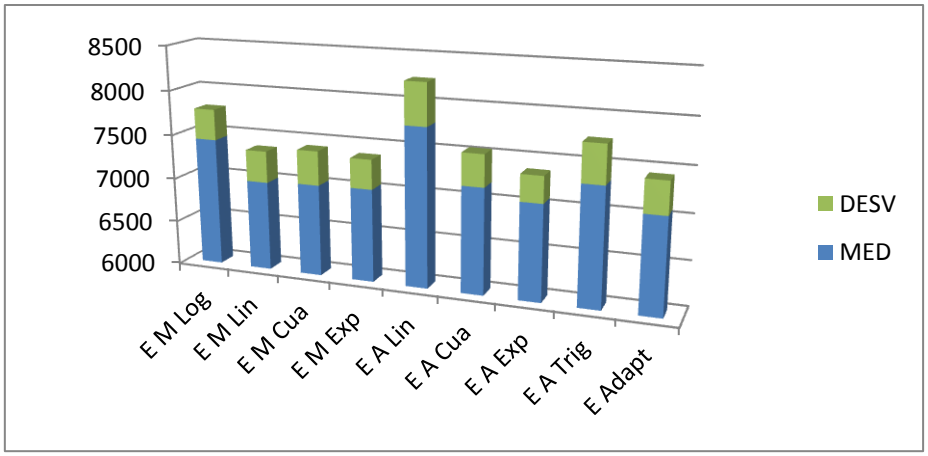

Figura 3.10: Comparación de resultados (objetivo) TSP 47

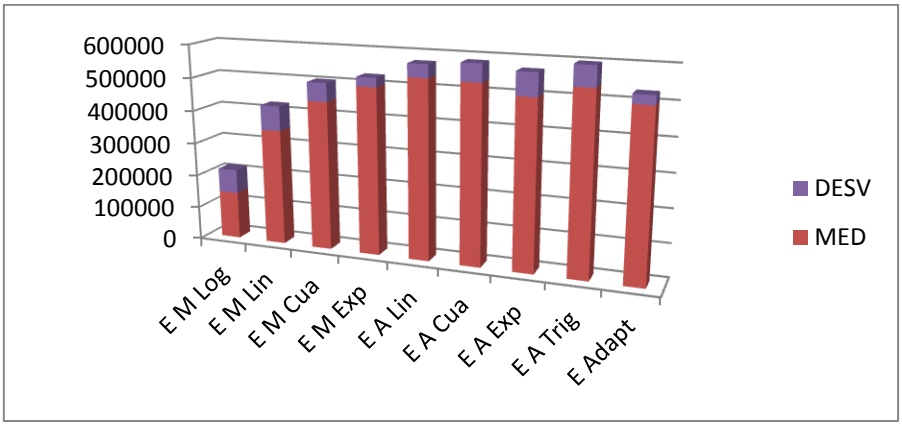

Figura 3.11: Comparación de resultados (nº iteraciones) TSP 47

		MED	DESV
E M Log	Objetivo	39105,32	2113,10
	Iteraciones	520860,56	418145,62
E M Lin	Objetivo	36109,17	1604,17
	Iteraciones	1252054,41	430911,30
E M Cua	Objetivo	35981,52	1719,84
	Iteraciones	1553160,7	276098,24
E M Exp	Objetivo	35334,72	1574,21
	Iteraciones	1761604,4	93123,80
E A Lin	Objetivo	38510,87	1971,90
	Iteraciones	1970423,64	271849,98
E A Cua	Objetivo	35134,61	1498,77
	Iteraciones	1935363,78	267271,02
E A Exp	Objetivo	35379,47	1536,45
	Iteraciones	1913526,41	387551,65
E A Trig	Objetivo	35536,52	1671,18
	Iteraciones	1811870,28	127674,83
E Adapt	Objetivo	35536,52	1671,18
	Iteraciones	1811870,28	127674,83

Tabla 3.3: Resultados TSP 80

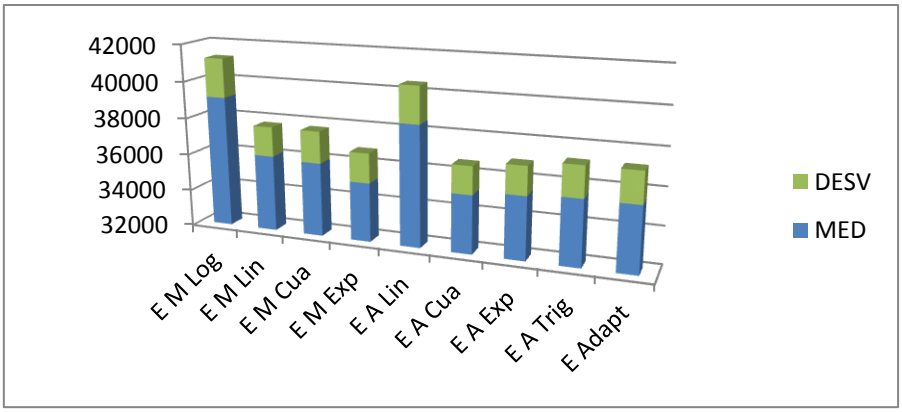

Figura 3.12: Comparación de resultados (objetivo) TSP 80

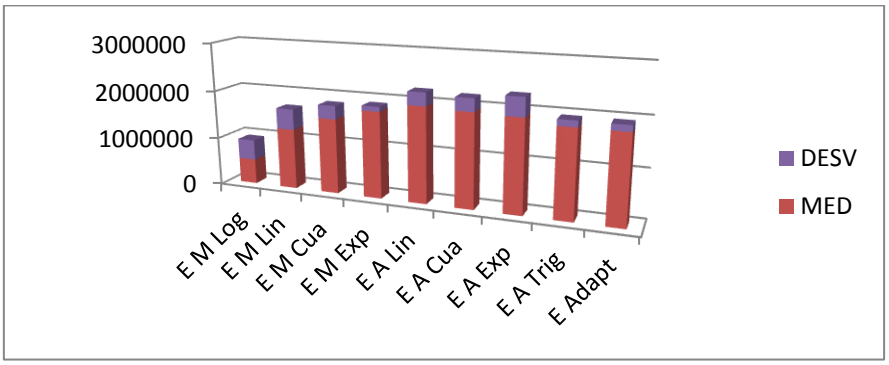

Figura 3.13: Comparación de resultados (nº iteraciones) TSP 80

		MED	DESV
E M Log	Objetivo	26633177,23	74803,05
	Iteraciones	203107,36	115788,34
E M Lin	Objetivo	26529682,05	52406,46
	Iteraciones	431308,09	124666,49
E M Cua	Objetivo	26520680,57	45713,60
	Iteraciones	502666,58	75805,55
E M Exp	Objetivo	26510487,17	41489,7
	Iteraciones	563650,78	38893,8
E A Lin	Objetivo	26603569,54	59799,74
	Iteraciones	625758,4	74479,14
E A Cua	Objetivo	26537607,08	46776,85
	Iteraciones	613327,98	75134,69
E A Exp	Objetivo	26542937,15	55483,35
	Iteraciones	513657,73	102397,37
E A Trig	Objetivo	26542883,95	50660,03
	Iteraciones	632113,28	94588,54
E Adapt	Objetivo	26507296,44	39920,66
	Iteraciones	559779,09	30921,78

Tabla 3.4: Resultados QAP 47

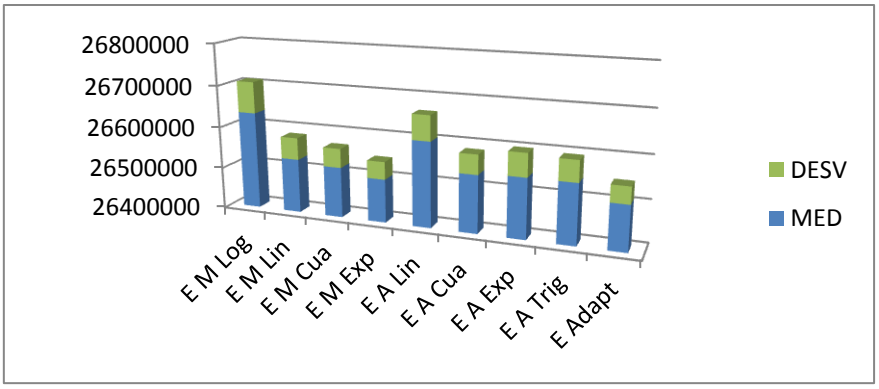

Figura 3.14: Comparación de resultados (objetivo) QAP 47

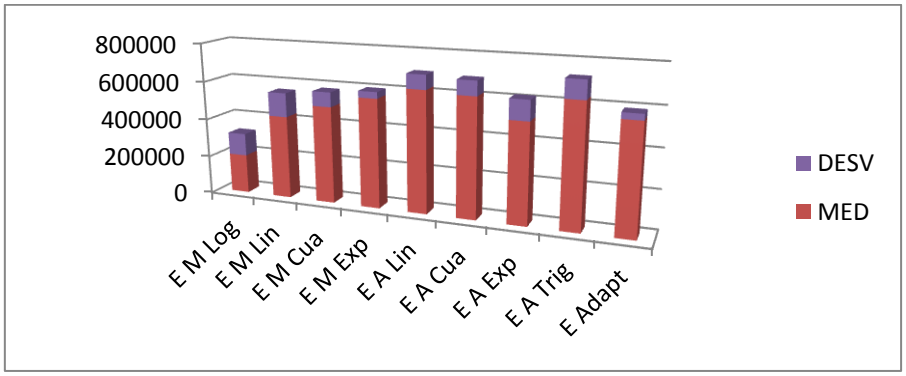

Figura 3.15: Comparación de resultados (nº iteraciones) QAP 47

		MED	DESV
E M Log	Objetivo	251827426,3	612601,80
	Iteraciones	541295,42	286643,42
E M Lin	Objetivo	250616081,8	346594,87
	Iteraciones	1153150,92	283266,83
E M Cua	Objetivo	250423922,9	320029,21
	Iteraciones	1413739,15	210670,69
E M Exp	Objetivo	250505166,5	323685,34
	Iteraciones	1661144,48	103933,62
E A Lin	Objetivo	251384049,3	484199,23
	Iteraciones	1839289,1	170599,30
E A Cua	Objetivo	250729377,7	384971,93
	Iteraciones	1790535,52	195289,55
E A Exp	Objetivo	250761021,1	374346,90
	Iteraciones	1472894,48	185779,66
E A Trig	Objetivo	250833251,8	369301,84
	Iteraciones	1802581,8	209695,33
E Adapt	Objetivo	250467485	362527,68
	Iteraciones	1677636,05	110480,80

Tabla 3.5: Resultados QAP 80

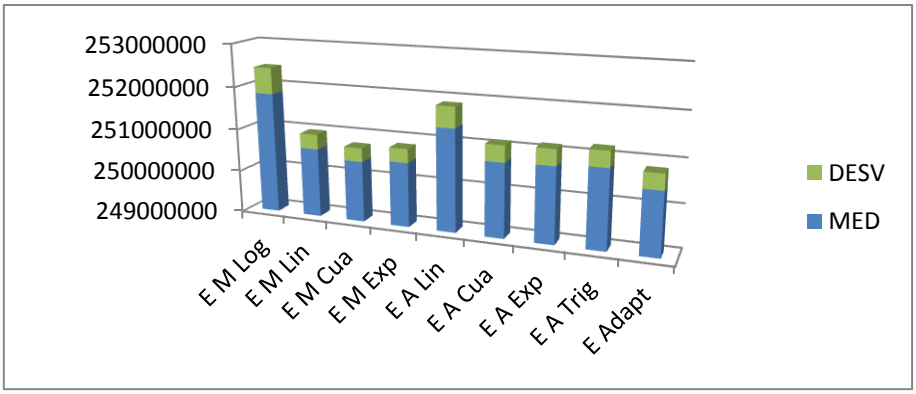

Figura 3.16: Comparación de resultados (objetivo) QAP 80

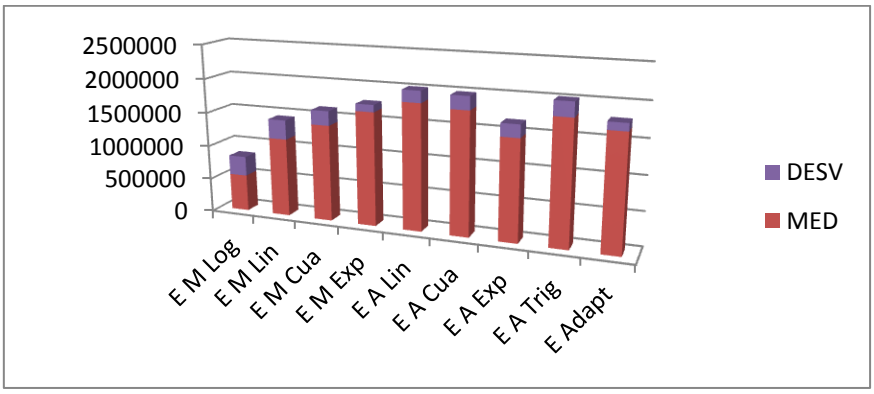

Figura 3.17: Comparación de resultados (nº iteraciones) QAP 80

En relación con la calidad del objetivo medio logrado y su desviación típica, puede observarse que el mejor comportamiento es proporcionado por cuatro programas:

- El enfriamiento multiplicativo exponencial (criterio estándar),
- el enfriamiento multiplicativo cuadrático,
- el enfriamiento aditivo exponencial, y
- el enfriamiento adaptativo.

Los peores resultados se obtienen con:

- El enfriamiento multiplicativo logarítmico y
- el enfriamiento aditivo lineal.

En relación con el número medio de iteraciones (tiempo de ejecución) y su desviación típica, puede observarse que todos los casos evaluados presentan un nivel similar, exceptuando el enfriamiento multiplicativo logarítmico que presenta una elevada desviación típica. Cabe mencionar que los métodos que mejor comportamiento proporcionan respecto a la desviación típica son el enfriamiento multiplicativo exponencial, y el enfriamiento adaptativo, que se basa en el anterior.

Combinando ambos aspectos, objetivo y tiempo de ejecución, puede concluirse que el mejor esquema de enfriamiento estudiado es el programa no monótono adaptativo basado en el propuesto por M. Locatelli [LOCA00], aunque sin diferencias significativas con respecto a los programas multiplicativo exponencial y multiplicativo cuadrático, tal como se comprueba en el análisis estadístico a continuación.

3.3.4. Análisis estadístico

A continuación, se van a realizar intervalos de confianza de la diferencia de medias – *para objetivos e iteraciones* – entre el enfriamiento exponencial estándar y cada una de las otras ocho estrategias de enfriamiento de la comparación, como técnica estadística para validar los resultados obtenidos en el apartado anterior.

Para analizar si puede inferirse una diferencia significativa entre *el enfriamiento multiplicativo exponencial y el enfriamiento multiplicativo logarítmico* se va a realizar un intervalo de confianza para la diferencia de medias, para lo cual se dispone de dos muestras aleatorias independientes de tamaño n_1 y n_2 respectivamente. El objetivo es comparar dos medias poblacionales μ_1 y μ_2. Para ello, lo adecuado es basarse en el estimador $\bar{x}_1 - \bar{x}_2$, diferencia entre ambas medias muestrales.

La situación que se estudia, en la comparación entre los diferentes algoritmos, es la de dos muestras aleatorias independientes del mismo tamaño y

suficientemente grandes, de manera que puede asegurarse la normalidad aproximada de ambas medias muestrales.

Se analizará si existe una diferencia significativa entre ambos algoritmos con respecto al coste del objetivo. Como las varianzas poblacionales son finitas y desconocidas, las sustituiremos por las cuasivarianzas muestrales S_1^2 y S_2^2, dando lugar al intervalo:

$$\left(-z_{\alpha/2} \leq Z \leq z_{\alpha/2} \right)$$

$$Z = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$$

A un nivel de confianza del 95%.

- Si $Z < -z_{\alpha/2}$, es mejor el algoritmo 1
- Si $Z > z_{\alpha/2}$, es mejor el algoritmo 2
- Si $\left(-z_{\alpha/2} \leq Z \leq z_{\alpha/2} \right)$, la prueba no permite afirmar que existan diferencias significativas entre ambos algoritmos

$\bar{x}_1$ = *media muestral (E M Exp)*

$\bar{x}_2$ = *media muestral (E M Log)*

n_1 = *tamaño muestra (E M Exp)*

n_2 = *tamaño muestra (E M Log)*

S_1 = *cuasivarianza muestral (E M Exp)*

S_2 = *cuasivarianza muestral (E M Log)*

Enfriamiento multiplicativo exponencial (E M Exp) y enfriamiento multiplicativo logarítmico (E M Log).

TSP 47:

		MED	DESV
E M Exp	Objetivo	7040,17	383,07

		MED	DESV
E M Log	Objetivo	7439,58	343,53

$$Z = -7{,}76 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
E M Exp	Objetivo	35334,72	1574,21

		MED	DESV
E M Log	Objetivo	39105,32	2113,10

$$Z = -14{,}24 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
E M Exp	Objetivo	26510487,2	41489,7

		MED	DESV
E M Log	Objetivo	26633177,23	74803,05

$$Z = -14{,}27 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Objetivo	250505167	323685,34

		MED	DESV
E M Log	Objetivo	251827426,3	612601,80

$$Z = -18{,}99 < -1{,}96 = -z_{\alpha/2}$$

Tabla 3.6: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo logarítmico

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp proporciona mejores soluciones que el algoritmo enfriamiento multiplicativo logarítmico E M Log.

En el caso de la diferencia de medias para las iteraciones:

TSP 47:

		MED	DESV
E M Exp	Iteraciones	498561,6	28723,4

		MED	DESV
E M Log	Iteraciones	142624,41	72210,02

$Z = 45{,}80 > 1{,}96 = z_{\alpha/2}$

TSP 80:

		MED	DESV
E M Exp	Iteraciones	1761604,4	93123,8

		MED	DESV
E M Log	Iteraciones	520860,56	418145,62

$Z = 28{,}96 > 1{,}96 = z_{\alpha/2}$

QAP 47:

		MED	DESV
E M Exp	Iteraciones	563650,78	38893,8

		MED	DESV
E M Log	Iteraciones	203107,36	115788,34

$Z = 29{,}51 > 1{,}96 = z_{\alpha/2}$

QAP 80:

		MED	DESV
E M Exp	Iteraciones	1661144,48	103933,6

		MED	DESV
E M Log	Iteraciones	541295,42	286643,42

$Z = 36{,}73 > 1{,}96 = z_{\alpha/2}$

Tabla 3.7: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo logarítmico

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo enfriamiento multiplicativo logarítmico E M Log converge más rápidamente que el algoritmo enfriamiento multiplicativo exponencial E M Exp.

En las comparativas, entre el enfriamiento multiplicativo exponencial y las restantes estrategias de enfriamiento, utilizaremos el mismo modelo de intervalo de confianza que en este primer caso.

Enfriamiento multiplicativo exponencial (E M Exp) y enfriamiento multiplicativo lineal (E M Lin).

TSP 47:

		MED	DESV
E M Exp	Objetivo	7040,17	383,07

		MED	DESV
E M Lin	Objetivo	7006	352,86

$$(-1{,}96 \leq 0{,}65 \leq 1{,}96)$$

TSP 80:

		MED	DESV
E M Exp	Objetivo	35334,72	1574,21

		MED	DESV
E M Lin	Objetivo	36109,17	1604,17

$$Z = -3{,}43 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
E M Exp	Objetivo	26510487,2	41489,7

		MED	DESV
E M Lin	Objetivo	26529682,05	52406,46

$$Z = -2{,}87 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Objetivo	250505167	323685,34

		MED	DESV
E M Lin	Objetivo	250616081,8	346594,87

$$Z = -2{,}34 < -1{,}96 = -z_{\alpha/2}$$

Tabla 3.8: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo lineal

- En TSP 47, la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos algoritmos.
- En TSP 80 y en las dos instancias del problema QAP se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp proporciona mejores soluciones que el algoritmo enfriamiento multiplicativo lineal E M Lin.

TSP 47:

		MED	DESV
E M Exp	Iteraciones	498561,6	28723,4

		MED	DESV
E M Lin	Iteraciones	349138,45	74278,16

$$Z = 18{,}76 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
E M Exp	Iteraciones	1761604,4	93123,8

		MED	DESV
E M Lin	Iteraciones	1252054,41	430911,29

$$Z = 11{,}5 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
E M Exp	Iteraciones	563650,78	38893,8

		MED	DESV
E M Lin	Iteraciones	431308,09	124666,49

$$Z = 10{,}13 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Iteraciones	1661144,48	103933,6

		MED	DESV
E M Lin	Iteraciones	1153150,92	283266,83

$$Z = 16{,}83 > 1{,}96 = z_{\alpha/2}$$

Tabla 3.9: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo lineal

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo enfriamiento multiplicativo lineal E M Lin converge más rápidamente que el algoritmo enfriamiento multiplicativo exponencial E M Exp.

Enfriamiento multiplicativo exponencial (E M Exp) y enfriamiento multiplicativo cuadrático (E M Cua).

TSP 47:

		MED	DESV
E M Exp	Objetivo	7040,17	383,07

		MED	DESV
E M Cua	Objetivo	7030,69	382,67

$$(-1{,}96 \leq 0{,}17 \leq 1{,}96)$$

TSP 80:

		MED	DESV
E M Exp	Objetivo	35334,72	1574,21

		MED	DESV
E M Cua	Objetivo	35981,52	1719,84

$$Z = -2{,}76 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
E M Exp	Objetivo	26510487,2	41489,7

		MED	DESV
E M Cua	Objetivo	26520680,57	45713,59

$$(-1{,}96 \leq -1{,}64 \leq 1{,}96)$$

QAP 80:

		MED	DESV
E M Exp	Objetivo	250505167	323685,34

		MED	DESV
E M Cua	Objetivo	250423922,9	320029,2

$$(-1{,}96 \leq 1{,}77 \leq 1{,}96)$$

Tabla 3.10: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo cuadrático

- En TSP 80, se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp proporciona mejores soluciones que el algoritmo enfriamiento multiplicativo cuadrático E M Cua.
- En TSP 47 y en las dos instancias del problema QAP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos algoritmos.

TSP 47:

		MED	DESV
E M Exp	Iteraciones	498561,6	28723,4

		MED	DESV
E M Cua	Iteraciones	447286,2	55776,8

$Z = 8{,}17 > 1{,}96 = z_{\alpha/2}$

TSP 80:

		MED	DESV
E M Exp	Iteraciones	1761604,4	93123,8

		MED	DESV
E M Cua	Iteraciones	1553160,7	276098,24

$Z = 7{,}15 > 1{,}96 = z_{\alpha/2}$

QAP 47:

		MED	DESV
E M Exp	Iteraciones	563650,78	38893,8

		MED	DESV
E M Cua	Iteraciones	502666,58	75805,55

$Z = 7{,}15 > 1{,}96 = z_{\alpha/2}$

QAP 80:

		MED	DESV
E M Exp	Iteraciones	1661144,48	103933,6

		MED	DESV
E M Cua	Iteraciones	1413739,15	210670,69

$Z = 10{,}53 > 1{,}96 = z_{\alpha/2}$

Tabla 3.11: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento multiplicativo cuadrático

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo enfriamiento multiplicativo cuadrático E M Cua converge más rápidamente que el algoritmo enfriamiento multiplicativo exponencial E M Exp.

Enfriamiento multiplicativo exponencial (E M Exp) y enfriamiento aditivo lineal (E A Lin).

TSP 47:

		MED	DESV
E M Exp	Objetivo	7040,17	383,06

		MED	DESV
E A Lin	Objetivo	7775,08	470,25

$$Z = -12{,}11 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
E M Exp	Objetivo	35334,72	1574,21

		MED	DESV
E A Lin	Objetivo	38510,87	1971,89

$$Z = -12{,}58 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
E M Exp	Objetivo	26510487,2	41489,7

		MED	DESV
E A Lin	Objetivo	26603569,54	59799,74

$$Z = -12{,}79 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Objetivo	250505167	323685,34

		MED	DESV
E A Lin	Objetivo	251384049,3	484199,23

$$Z = -15{,}01 < -1{,}96 = -z_{\alpha/2}$$

Tabla 3.12: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo lineal

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp proporciona mejores soluciones que el algoritmo enfriamiento aditivo lineal E A Lin.

TSP 47:

		MED	DESV
E M Exp	Iteraciones	498561,6	28723,4

		MED	DESV
E A Lin	Iteraciones	535389,66	39308,50

$$Z = -7{,}56 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
E M Exp	Iteraciones	1761604,4	93123,8

		MED	DESV
E A Lin	Iteraciones	1970423,64	271849,97

$$Z = -7{,}23 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
E M Exp	Iteraciones	563650,78	38893,8

		MED	DESV
E A Lin	Iteraciones	625758,4	74479,14

$$Z = -7{,}35 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Iteraciones	1661144,48	103933,6

		MED	DESV
E A Lin	Iteraciones	1839289,1	170599,3

$$Z = -8{,}87 < -1{,}96 = -z_{\alpha/2}$$

Tabla 3.13: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo lineal

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp converge más rápidamente que el algoritmo enfriamiento aditivo lineal E A Lin.

Enfriamiento multiplicativo exponencial (E M Exp) y enfriamiento aditivo cuadrático (E A Cua).

TSP 47:

		MED	DESV
E M Exp	Objetivo	7040,17	383,06

		MED	DESV
E A Cua	Objetivo	7183,73	355,62

$$Z = -2{,}74 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
E M Exp	Objetivo	35334,72	1574,21

		MED	DESV
E A Cua	Objetivo	35134,61	1498,77

$$(-1{,}96 \leq 0{,}92 \leq 1{,}96)$$

QAP 47:

		MED	DESV
E M Exp	Objetivo	26510487,2	41489,7

		MED	DESV
E A Cua	Objetivo	26537607,08	46776,85

$$Z = -4{,}31 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Objetivo	250505167	323685,34

		MED	DESV
E A Cua	Objetivo	250729377,7	384971,93

$$Z = -4{,}43 < -1{,}96 = -z_{\alpha/2}$$

Tabla 3.14: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo cuadrático

- En TSP 80, la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos algoritmos.
- En TSP 47 y en las dos instancias del problema QAP se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp proporciona mejores soluciones que el algoritmo enfriamiento aditivo cuadrático E A Cua.

TSP 47:

		MED	DESV
E M Exp	Iteraciones	498561,6	28723,4

		MED	DESV
E A Cua	Iteraciones	532664,19	52723,84

$$Z = -5{,}68 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
E M Exp	Iteraciones	1761604,4	93123,8

		MED	DESV
E A Cua	Iteraciones	1935363,78	267271,02

$$Z = -6{,}10 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
E M Exp	Iteraciones	563650,78	38893,8

		MED	DESV
E A Cua	Iteraciones	613327,98	75134,69

$$Z = -5{,}84 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Iteraciones	1661144,48	103933,6

		MED	DESV
E A Cua	Iteraciones	1790535,52	195289,55

$$Z = -5{,}82 < -1{,}96 = -z_{\alpha/2}$$

Tabla 3.15: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo cuadrático

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp converge más rápidamente que el algoritmo enfriamiento aditivo exponencial E A Cua.

Enfriamiento multiplicativo exponencial (E M Exp) y enfriamiento aditivo exponencial (E A Exp).

TSP 47:

		MED	DESV
E M Exp	Objetivo	7040,17	383,06

		MED	DESV
E A Exp	Objetivo	7072,56	293,97

$$(-1{,}96 \leq -0{,}67 \leq 1{,}96)$$

TSP 80:

		MED	DESV
E M Exp	Objetivo	35334,72	1574,21

		MED	DESV
E A Exp	Objetivo	35379,47	1536,45

$$(-1{,}96 \leq -0{,}20 \leq 1{,}96)$$

QAP 47:

		MED	DESV
E M Exp	Objetivo	26510487,2	41489,7

		MED	DESV
E A Exp	Objetivo	26542937,15	55483,35

$$Z = -4{,}66 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Objetivo	250505167	323685,343

		MED	DESV
E A Exp	Objetivo	250761021,1	374346,8

$$Z = -5{,}14 < -1{,}96 = -z_{\alpha/2}$$

Tabla 3.16: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo exponencial

- En las dos instancias del problema TSP, la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos algoritmos.
- En las dos instancias del problema QAP se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp proporciona mejores soluciones que el algoritmo enfriamiento aditivo exponencial E A Exp.

TSP 47:

		MED	DESV
E M Exp	Iteraciones	498561,6	28723,4

		MED	DESV
E A Exp	Iteraciones	502794,07	67931,19

$$(-1{,}96 \leq -0{,}57 \leq 1{,}96)$$

TSP 80:

		MED	DESV
E M Exp	Iteraciones	1761604,4	93123,8

		MED	DESV
E A Exp	Iteraciones	1913526,41	387551,65

$$Z = -3{,}79 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
E M Exp	Iteraciones	563650,78	38893,8

		MED	DESV
E A Exp	Iteraciones	513657,73	102397,37

$$Z = 4{,}54 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Iteraciones	1661144,48	103933,6

		MED	DESV
E A Exp	Iteraciones	1472894,48	185779,66

$$Z = 8{,}84 > 1{,}96 = z_{\alpha/2}$$

Tabla 3.17: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo exponencial

- En TSP 47, la prueba no permite afirmar que existan diferencias significativas en velocidad de convergencia entre ambos algoritmos.
- En TSP 80 se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp converge más rápidamente que el algoritmo enfriamiento aditivo exponencial E A Exp.
- En las dos instancias del problema QAP se puede considerar que el algoritmo enfriamiento aditivo exponencial E A Exp converge más rápidamente que el algoritmo enfriamiento multiplicativo exponencial E M Exp.

Enfriamiento multiplicativo exponencial (E M Exp) y enfriamiento aditivo trigonométrico (E A Trig).

TSP 47:

		MED	DESV
E M Exp	Objetivo	7040,17	383,06

		MED	DESV
E A Trig	Objetivo	7324,19	425,62

$$Z = -4{,}93 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
E M Exp	Objetivo	35334,72	1574,21

		MED	DESV
E A Trig	Objetivo	35536,52	1671,18

$$\left(-1{,}96 \leq -0{,}87 \leq 1{,}96\right)$$

QAP 47:

		MED	DESV
E M Exp	Objetivo	26510487,2	41489,7

		MED	DESV
E A Trig	Objetivo	26542883,95	50660,03

$$Z = -4{,}92 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Objetivo	250505167	323685,343

		MED	DESV
E A Trig	Objetivo	250833251,8	369301,848

$$Z = -6{,}64 < -1{,}96 = -z_{\alpha/2}$$

Tabla 3.18: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo trigonométrico

- En TSP 80, la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos algoritmos.
- En TSP 47 y en las dos instancias del problema QAP se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp proporciona mejores soluciones que el algoritmo enfriamiento aditivo trigonométrico E A Trig.

TSP 47:

		MED	DESV
E M Exp	Iteraciones	498561,6	28723,4

		MED	DESV
E A Trig	Iteraciones	538035,36	60389,06

$$Z = -5{,}87 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
E M Exp	Iteraciones	1761604,4	93123,8

		MED	DESV
E A Trig	Iteraciones	1811870,28	127674,83

$$Z = -3{,}16 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
E M Exp	Iteraciones	563650,78	38893,8

		MED	DESV
E A Trig	Iteraciones	632113,28	94588,53

$$Z = -6{,}66 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
E M Exp	Iteraciones	1661144,48	103933,6

		MED	DESV
E A Trig	Iteraciones	1802581,8	209695,32

$$Z = -6{,}01 < -1{,}96 = -z_{\alpha/2}$$

Tabla 3.19: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento aditivo trigonométrico

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp converge más rápidamente que el algoritmo enfriamiento aditivo trigonométrico E A Trig.

Enfriamiento multiplicativo exponencial (E M Exp) y enfriamiento no monótono adaptativo (E Adapt).

TSP 47:

		MED	DESV
E M Exp	Objetivo	7040,17	383,07

		MED	DESV
E Adapt	Objetivo	7073,42	359,56

$$(-1{,}96 \leq -0{,}63 \leq 1{,}96)$$

TSP 80:

		MED	DESV
E M Exp	Objetivo	35334,72	1574,21

		MED	DESV
E Adapt	Objetivo	35536,52	1671,18

$$(-1{,}96 \leq -0{,}87 \leq 1{,}96)$$

QAP 47:

		MED	DESV
E M Exp	Objetivo	26510487,2	41489,7

		MED	DESV
E Adapt	Objetivo	26507296,44	39920,66

$$(-1{,}96 \leq 0{,}55 \leq 1{,}96)$$

QAP 80:

		MED	DESV
E M Exp	Objetivo	250505167	323685,34

		MED	DESV
E Adapt	Objetivo	250467485	362527,68

$$(-1{,}96 \leq 0{,}77 \leq 1{,}96)$$

Tabla 3.20: Intervalo de confianza para la diferencia de medias para objetivos entre el enfriamiento multiplicativo exponencial y enfriamiento no monótono adaptativo

- En las dos instancias de los problemas TSP y QAP, la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos algoritmos.

TSP 47:

		MED	DESV
E M Exp	Iteraciones	498561,6	28723,4

		MED	DESV
E Adapt	Iteraciones	502653,87	26854,95

$$(-1{,}96 \leq -1{,}03 \leq 1{,}96)$$

TSP 80:

		MED	DESV
E M Exp	Iteraciones	1761604,4	93123,8

		MED	DESV
E Adapt	Iteraciones	1811870,28	127674,83

$$Z = -3{,}16 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
E M Exp	Iteraciones	563650,78	38893,8

		MED	DESV
E Adapt	Iteraciones	559779,09	30921,78

$$(-1{,}96 \leq 0{,}77 \leq 1{,}96)$$

QAP 80:

		MED	DESV
E M Exp	Iteraciones	1661144,48	103933,6

		MED	DESV
E Adapt	Iteraciones	1677636,05	110480,80

$$(-1{,}96 \leq -1{,}08 \leq 1{,}96)$$

Tabla 3.21: Intervalo de confianza para la diferencia de medias para iteraciones entre el enfriamiento multiplicativo exponencial y enfriamiento no monótono adaptativo

- En TSP 80, se puede considerar que el algoritmo enfriamiento multiplicativo exponencial E M Exp converge más rápidamente que el algoritmo enfriamiento no monótono adaptativo E Adapt.
- En TSP 47 y en las dos instancias del problema QAP la prueba no permite afirmar que existan diferencias significativas en velocidad de convergencia entre ambos algoritmos.

A continuación se recoge en forma resumida la información de la comparativa en sendas tablas, una sobre objetivos y otra sobre convergencia temporal.

En la tabla 3.22, el símbolo '+' significa que el enfriamiento estándar E M Exp da mejores soluciones, '=' quiere decir que la diferencia no es significativa, y '-' significa que el enfriamiento E M Exp ofrece soluciones de peor calidad.

Instancia	E M Log	E M Lin	E M Cua	E A Lin	E A Cua	E A Exp	E A Trig	E Adapt
TSP 47	+	=	=	+	+	=	+	=
TSP 80	+	+	+	+	=	=	=	=
QAP 47	+	+	=	+	+	+	+	=
QAP 80	+	+	=	+	+	+	+	=

Tabla 3.22: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de objetivos según la distribución normal

En la tabla 3.23, '+' significa que el enfriamiento estándar E M Exp converge más rápido, '=' quiere decir que la diferencia no es significativa, y '-' significa que el enfriamiento E M Exp converge más lentamente.

Instancia	E M Log	E M Lin	E M Cua	E A Lin	E A Cua	E A Exp	E A Trig	E Adapt
TSP 47	-	-	-	+	+	=	+	=
TSP 80	-	-	-	+	+	+	+	+
QAP 47	-	-	-	+	+	-	+	=
QAP 80	-	-	-	+	+	-	+	=

Tabla 3.23: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de número de iteraciones según la distribución normal

3.3.5. Análisis de resultados

Si se contrastan los resultados con las curvas de reducción de la temperatura de cada programa de enfriamiento, tomando siempre como referencia la forma de la curva del enfriamiento multiplicativo exponencial, es posible establecer las siguientes conjeturas explicativas:

- La mala calidad del objetivo medio obtenido en el enfriamiento multiplicativo logarítmico se debe a que la temperatura desciende hasta un valor bajo de forma brusca en los ciclos iniciales, para luego continuar reduciéndose de forma muy suave. Debido a ello, en la mayor parte del proceso las transiciones de estado se realizan a baja temperatura. Esta baja temperatura sólo permite aceptar pequeños saltos hacia soluciones de peor calidad, lo que reduce notablemente la capacidad del algoritmo del recocido simulado para escapar de los óptimos locales. Además, como la temperatura llega rápidamente hasta un nivel cercano al de alta probabilidad de terminación del algoritmo y se mantiene en ese nivel reduciéndose lentamente durante buena parte de la ejecución, la terminación puede producirse prematuramente o llegar a retrasarse muchos ciclos, lo que explica que este enfriamiento posea una gran variabilidad respecto al número total de iteraciones de procesamiento.

- La mala calidad del enfriamiento aditivo lineal puede deberse a que la reducción de la temperatura en la parte final del procesamiento es demasiado fuerte, pues la pendiente de la curva de descenso es constante, mientras que en los restantes modelos de enfriamiento en esta fase la pendiente se suaviza notablemente.

La regularidad de los restantes esquemas de enfriamiento, tanto en el objetivo logrado como en el tiempo de ejecución, se debe a que la reducción de la temperatura se produce de forma moderada en las partes inicial y central del procesamiento, y más suave en la parte final del mismo, lo que permite al algoritmo desarrollar plenamente su potencialidad de escape de óptimos locales.

Las principales conclusiones que pueden extraerse de la comparativa realizada entre programas de enfriamiento del algoritmo meta-heurístico del recocido simulado son:

- Con respecto al nivel de calidad del objetivo y su relación con la forma de la curva de decrecimiento de la temperatura, puede afirmarse que el algoritmo del recocido simulado funciona bien respecto a su capacidad de escape de óptimos locales cuando la curva tiene pendiente moderada en las partes inicial y central del procesamiento, y más suave en la parte final del mismo, tal como sucede en el enfriamiento multiplicativo exponencial estándar. La forma concreta (convexa, sigmoidal) de la curva en las partes inicial y central no parece relevante, pero en la parte final una curva de descenso más suave puede favorecer mejores niveles de optimización.

- Con respecto al tiempo de ejecución, todos los programas de enfriamiento estudiados para el recocido simulado presentan un coste similar, debido – lógicamente – a que sus parámetros han sido configurados para asemejarse en este comportamiento al enfriamiento multiplicativo exponencial de referencia. Sin embargo, la desviación típica del número de iteraciones efectuadas parece estar en relación con la cola de la curva de reducción de la temperatura en la fase final del algoritmo. Una cola inversamente logarítmica produce un descenso final de la temperatura más suave y de mayor desviación, mientras que las colas de tipo inversamente lineal o cuadrático generan menor desviación. Precisamente el enfriamiento multiplicativo exponencial y el enfriamiento adaptativo – basado en el anterior –, poseen una cola que se anula más rápidamente y proporcionan los mejores valores de desviación temporal del algoritmo.

Capítulo 4: Enfriamiento Adaptativo

“No nos atrevemos a muchas cosas
porque son difíciles, pero son difíciles
porque no nos atrevemos a hacerlas”

Lucio Anneo Séneca

El recocido simulado se caracteriza por sus propiedades de convergencia hacia soluciones de alta calidad pero con un elevado tiempo computacional.

Con esta premisa, esta investigación se centra en reducir el tiempo de ejecución del algoritmo manteniendo el coste del objetivo. En el capítulo anterior se realizó un análisis previo de las funciones de actualización de la temperatura, llegándose a la conclusión de que el enfriamiento multiplicativo exponencial, el estándar propuesto por Kirkpatrick, Gelatt y Vecchi [KIRK83], es uno de los mejores, sin diferencias estadísticamente significativas en eficiencia respecto a las restantes mejores funciones de enfriamiento contrastadas. Teniendo esto en cuenta, a continuación se van a realizar tres estudios diferentes sobre la influencia de los parámetros del programa de enfriamiento en el comportamiento del algoritmo, utilizando como referencia el enfriamiento multiplicativo exponencial, y ejecutando las pruebas con las instancias de los problemas TSP y QAP presentadas en el capítulo 3. Estos estudios son:

1. *La influencia de la temperatura inicial en el comportamiento del algoritmo*.
2. *La influencia del parámetro* α *cuando éste se modifica dinámicamente en la ecuación de reducción exponencial de la temperatura*.

3. *La influencia combinada de la temperatura inicial y el parámetro* α *en la eficiencia del algoritmo.*

4.1. INFLUENCIA DE LA TEMPERATURA INICIAL EN LA EFICIENCIA DEL ALGORITMO

El objeto del estudio que se presenta a continuación es analizar *la influencia de la temperatura inicial en el comportamiento del algoritmo* usando como referencia el enfriamiento multiplicativo exponencial. En virtud de la analogía con el proceso físico del recocido y, de acuerdo con el algoritmo estándar propuesto por Kirkpatrick, Gelatt y Vecchi [KIRK83], se sabe que la temperatura inicial del sistema debe ser suficientemente elevada como para aceptar cualquier transición de estado que suponga una pérdida del coste del objetivo en curso. El planteamiento es valorar el comportamiento del algoritmo si se inicializa la temperatura con un valor que suponga la ejecución de un porcentaje entre el 5% y el 90% del número medio de ciclos de enfriamiento que se realizan cuando se parte de la temperatura inicial estándar.

4.1.1. Elección de parámetros de enfriamiento

Para obtener el valor inicial de la temperatura T_0 en función del número de ciclos que vayan a ejecutarse, emplearemos la siguiente fórmula:

$$\bar{T} + \sigma = T_0 \alpha^{np} \Rightarrow T_0 = \frac{\bar{T} + \sigma}{\alpha^{np}}$$

donde:

- T_0 es la temperatura inicial a calcular.
- α es el factor constante de decremento de la temperatura tras cada ciclo de enfriamiento, $\alpha = 0'95$.

- $\overline{T}$ es el valor medio de la temperatura final para la instancia del problema considerado.
- σ es la desviación típica de la temperatura final para la instancia del problema considerado.
- *n* es el número medio de ciclos de enfriamiento para la instancia del problema considerado, partiendo de la temperatura inicial estándar.
- *p* es el tanto por uno del número de ciclos de enfriamiento que se desean efectuar, siendo $p = 0'05\,(5\%), 0'1 (10\%), 0'2\,(20\%),..., 0'9\,(90\%)$.

En las pruebas realizadas en este estudio se han considerado los valores medios de $\overline{T}$, σ y *n* obtenidos en la comparativa del capítulo 3 para el enfriamiento multiplicativo exponencial sobre las instancias TSP 47, TSP 80, QAP 47, y QAP 80. Sin embargo, con vistas a la implementación práctica de programas de enfriamiento con reducción de la temperatura inicial para otros problemas y/o instancias, estos valores pueden aproximarse fácilmente (sin ejecuciones previas del recocido simulado) de la siguiente forma:

- La temperatura final $\overline{T} + \sigma$ a la que se llega en el algoritmo del recocido simulado está determinada por la función de Boltzmann $e^{\frac{-|f(\boldsymbol{x})-f(\boldsymbol{y})|}{T}}$, donde ***x*** e ***y*** son dos soluciones candidatas viables en comparación, *f* es la función objetivo y *T* es la temperatura del sistema. Cuando la función de Boltzmann toma valores próximos a cero, según el criterio de Metropolis se rechazan todas las soluciones de peor calidad a la actual, con lo que el algoritmo se comporta como una búsqueda local simple, terminando su ejecución en ese ciclo de enfriamiento. Así, la temperatura final $\overline{T} + \sigma$ puede estimarse con un algoritmo esencialmente idéntico al presentado en el capítulo 3 para el cálculo de la temperatura inicial. En este caso, se repiten ensayos de 10 transiciones de estado hacia soluciones de peor calidad, en los que en lugar de incrementar la temperatura tras cada ensayo, la temperatura se decrementa multiplicándola por un factor β menor

que la unidad, siendo usualmente $0'67 \leq \beta \leq 0'9$, y así hasta que todas las transiciones de estado de un ensayo sean rechazadas. También puede estimarse de acuerdo con la fórmula:

$$e^{\frac{-\sup \Delta f}{\overline{T}+\sigma}} = P(\boldsymbol{y}) \Rightarrow \overline{T}+\sigma = \frac{-\sup \Delta f}{\ln(P(\boldsymbol{y}))}$$

donde $P(\boldsymbol{y})$ es una probabilidad de aceptación de soluciones peores cercana a cero (por ejemplo, 0.01), y sup Δf es una cota superior del incremento medio de coste entre soluciones viables del problema, el cual puede obtenerse a partir de un pequeño conjunto de soluciones candidatas viables elegidas al azar.

- El número n de ciclos de enfriamiento para el enfriamiento multiplicativo exponencial, conocidas la temperatura inicial estándar T_0 y la temperatura final $\overline{T}+\sigma$, puede calcularse con la fórmula:

$$\overline{T}+\sigma = T_0 \alpha^n \Rightarrow n = \frac{\ln\left(\frac{\overline{T}+\sigma}{T_0}\right)}{\ln(\alpha)}$$

La tabla 4.1 muestra los valores de la temperatura inicial cuando se consideran porcentajes entre el 5% y el 90% del número de ciclos de enfriamiento correspondientes a la temperatura inicial estándar.

	TSP 47	**TSP 80**	**QAP 47**	**QAP 80**
5%	13,62	9,55	1041,11	3782,74
10%	19,81	15,19	1573,33	5805,17
20%	41,89	38,43	3593,11	13671,96
30%	88,59	97,25	8205,79	32199,3
40%	187,33	246,1	18740,04	75833,71
50%	396,14	622,76	42797,7	178598,61
60%	837,69	1575,89	97739,57	420623,83
70%	1771,40	3987,76	223213,45	990625,88
80%	3745,87	10090,96	509765,36	2333057,60
90%	7921,15	25534,96	1164180,38	5494665,43

Tabla 4.1: Valor de T_0 según el nº ciclos que se ejecutan

4.1.2. Resultados de las pruebas

Las tablas 4.2 – 4.5, figuras 4.1 – 4.8 muestran los resultados de las pruebas para los diferentes porcentajes del número de ciclos de enfriamiento utilizados en cada instancia de los problemas TSP y QAP.

Para cada instancia de los problemas se han realizado 100 ejecuciones con el programa de enfriamiento multiplicativo exponencial, obteniéndose el valor medio y la desviación típica tanto del objetivo logrado como del número total de iteraciones empleadas en cada caso.

		MED	DESV
T_0 = 5%	Objetivo	8210,16	617,29
	Iteraciones	46185,08	24720,54
T_0 = 10%	Objetivo	7983,96	510,31
	Iteraciones	61175,17	22889,61
T_0 = 20%	Objetivo	7479,11	417,95
	Iteraciones	103836,86	26581,96
T_0 = 30%	Objetivo	7103,36	329,96
	Iteraciones	152042,16	25813,36
T_0 = 40%	Objetivo	7040,21	442,63
	Iteraciones	200192,23	22828,81
T_0 = 50%	Objetivo	7045,54	348,42
	Iteraciones	251902	24670,38
T_0 = 60%	Objetivo	7005,08	418,25
	Iteraciones	302843,56	25129,52
T_0 = 70%	Objetivo	7063,76	363,37
	Iteraciones	348378,66	24987,89
T_0 = 80%	Objetivo	6970,64	392,03
	Iteraciones	397578,08	26025,19
T_0 = 90%	Objetivo	6964,83	390,59
	Iteraciones	449647,33	23545,22
T_0 = 100%	Objetivo	7040,17	383,06
	Iteraciones	498561,6	28723,43

Tabla 4.2: Resultados TSP 47

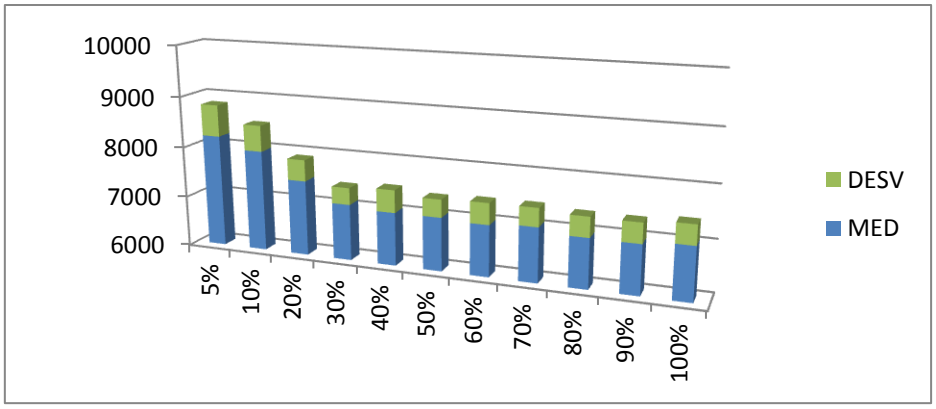

Figura 4.1: Comparación de resultados (objetivo) TSP 47

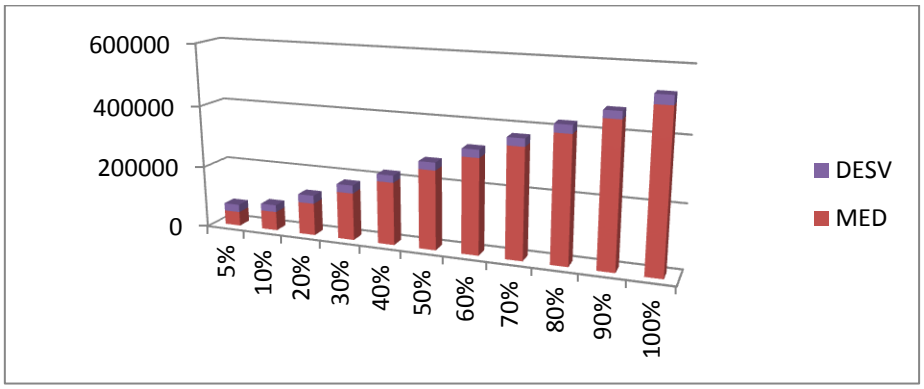

Figura 4.2: Comparación de resultados (nº iteraciones) TSP 47

		MED	DESV
T_0 = 5%	Objetivo	41710,48	2750,48
	Iteraciones	136325,13	79539,82
T_0 = 10%	Objetivo	40662,46	2528,46
	Iteraciones	214508,21	94145,59
T_0 = 20%	Objetivo	39339,81	1881,37
	Iteraciones	401596,45	125630,30
T_0 = 30%	Objetivo	37686,68	1804,39
	Iteraciones	544036,7	105355,24
T_0 = 40%	Objetivo	35668,78	1605,24
	Iteraciones	724661,95	111532,25
T_0 = 50%	Objetivo	35458,71	1618,69
	Iteraciones	919739,93	117770,42
T_0 = 60%	Objetivo	35389,53	1707,95
	Iteraciones	1102722,29	105368,56
T_0 = 70%	Objetivo	35400,44	1659,42
	Iteraciones	1260857,94	115740,23
T_0 = 80%	Objetivo	35297,91	1592,95
	Iteraciones	1458247,74	94025,93
T_0= 90%	Objetivo	35666,47	1467,09
	Iteraciones	1611305,03	105684,16
T_0 = 100%	Objetivo	35334,72	1574,20
	Iteraciones	1761604,4	93123,83

Tabla 4.3: Resultados TSP 80

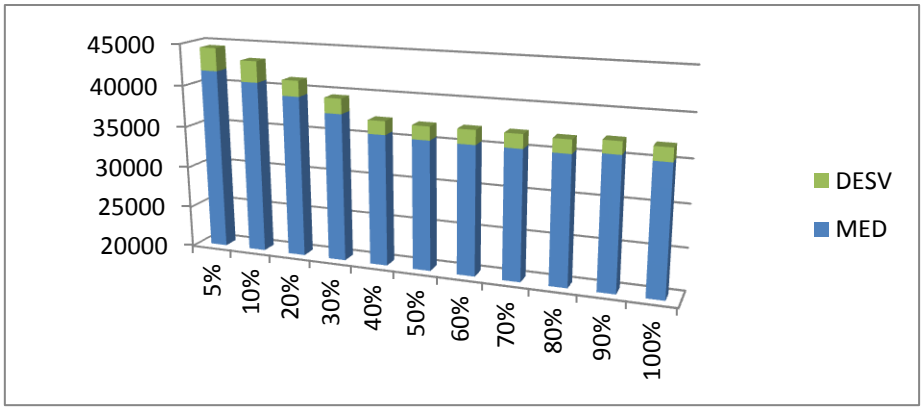

Figura 4.3: Comparación de resultados (objetivo) TSP 80

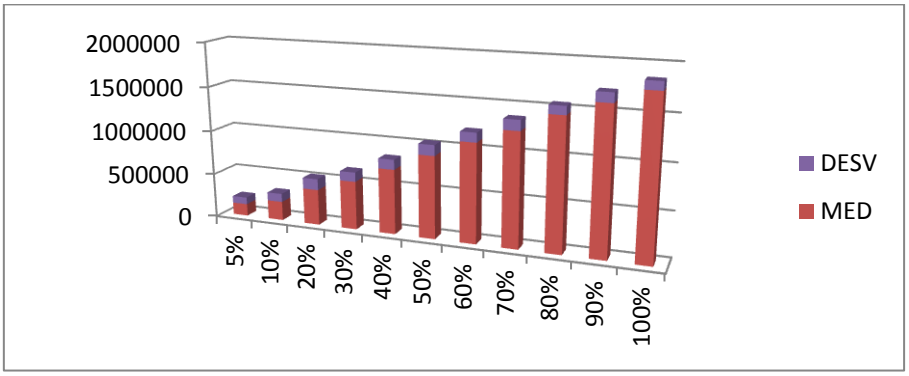

Figura 4.4: Comparación de resultados (nº iteraciones) TSP 80

		MED	DESV
T_0 = 5%	Objetivo	26698708,9	93314,03
	Iteraciones	56104,25	31642,79
T_0 = 10%	Objetivo	26666354,57	82103,48
	Iteraciones	81298,2	38283,01
T_0 = 20%	Objetivo	26629803,4	76495,26
	Iteraciones	131972,95	39164,56
T_0 = 30%	Objetivo	26589921,47	73187,13
	Iteraciones	187025,3	40043,07
T_0 = 40%	Objetivo	26526871,8	44315,1
	Iteraciones	235992,51	39935,80
T_0 = 50%	Objetivo	26507920,5	45163,54
	Iteraciones	287133,05	35910,29
T_0 = 60%	Objetivo	26509624,83	39781,98
	Iteraciones	347275,92	42963,01
T_0 = 70%	Objetivo	26518716,55	44233,08
	Iteraciones	405089,88	40466,83
T_0 = 80%	Objetivo	26514036,58	37499,35
	Iteraciones	455730,19	41736,36
T_0 = 90%	Objetivo	26523177,45	46896,42
	Iteraciones	512481,52	38925,23
T_0 = 100%	Objetivo	26510487,17	41489,66
	Iteraciones	563650,78	38893,80

Tabla 4.4: Resultados QAP 47

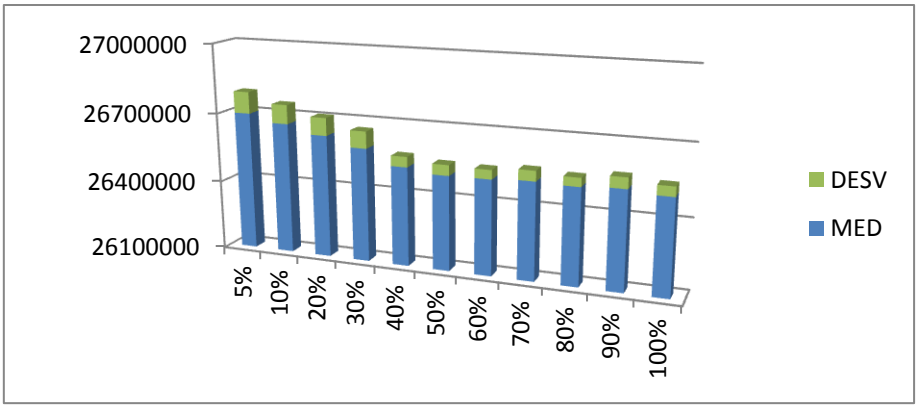

Figura 4.5: Comparación de resultados (objetivo) QAP 47

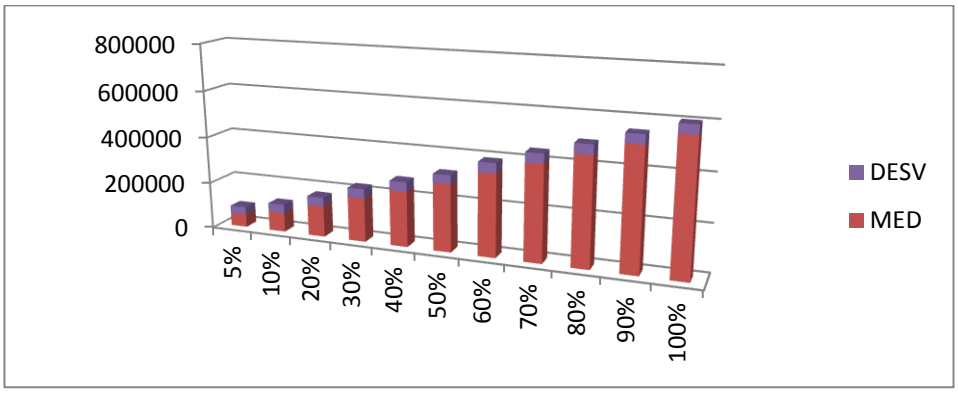

Figura 4.6: Comparación de resultados (nº iteraciones) QAP 47

		MED	DESV
T_0 = 5%	Objetivo	252415960,4	670898,60
	Iteraciones	170431,42	93327,66
T_0 = 10%	Objetivo	252323083,9	670236,78
	Iteraciones	202831,24	76603,39
T_0 = 20%	Objetivo	251816859,3	679054,76
	Iteraciones	374184,51	93821,46
T_0 = 30%	Objetivo	251518934,4	544294,80
	Iteraciones	530403,66	94148,30
T_0 = 40%	Objetivo	250658076,3	364640,68
	Iteraciones	689628,73	98553,56
T_0 = 50%	Objetivo	250501889,1	333451,58
	Iteraciones	848364,73	86182,36
T_0 = 60%	Objetivo	250503326,5	351551,79
	Iteraciones	1017084,1	101138,79
T_0 = 70%	Objetivo	250580944,8	329969,12
	Iteraciones	1192575,18	100023,27
T_0 = 80%	Objetivo	250541184,8	325238,13
	Iteraciones	1349386,96	102576,62
T_0 = 90%	Objetivo	250453507	387819,59
	Iteraciones	1510488,99	121394,56
T_0 = 100%	Objetivo	250574322,1	335723,84
	Iteraciones	1663308,76	100600,05

Tabla 4.5: Resultados QAP 80

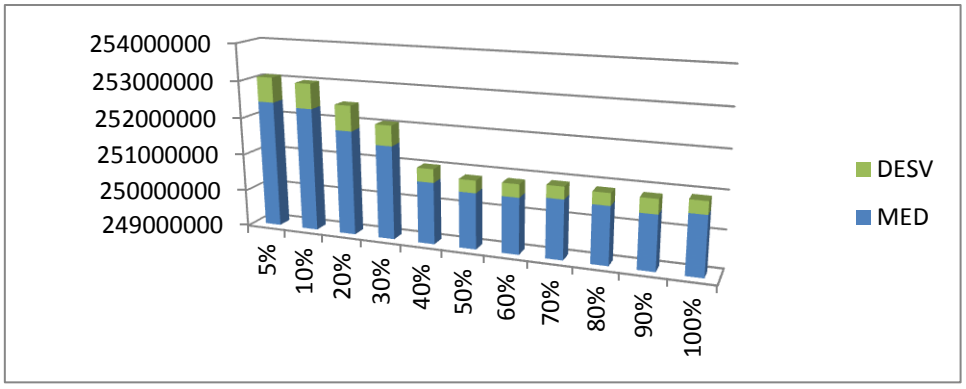

Figura 4.7: Comparación de resultados (objetivo) QAP 80

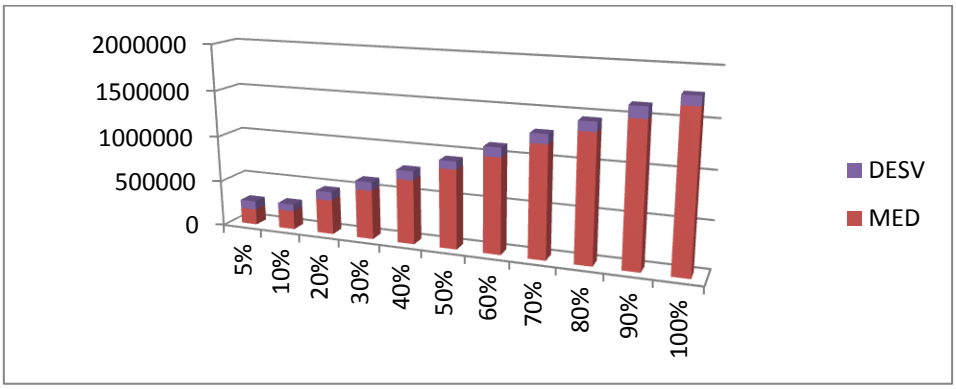

Figura 4.8: Comparación de resultados (nº iteraciones) QAP 80

4.1.3. Análisis de los resultados

A continuación, se va a evaluar el comportamiento del algoritmo en los diferentes escenarios planteados – cuando el algoritmo se inicializa con un porcentaje entre el 5% y el 90% del número de ciclos de la temperatura inicial estándar –.

Prueba estadística

Se va a determinar con qué porcentaje (5%, 10%, 20%,..., 90%) la calidad de la solución media obtenida (objetivo) no es significativamente diferente (peor) que la obtenida con la temperatura inicial estándar. Para ello, se va a realizar una estimación estadística por intervalo al 95% de confianza para la diferencia de medias, tomando como referencia el objetivo medio obtenido con el enfriamiento correspondiente a la temperatura inicial estándar.

Para analizar si puede inferirse una diferencia significativa se va a realizar un intervalo de confianza para la diferencia de medias, para lo cual se dispone de dos muestras aleatorias independientes de tamaño n_1 y n_2 respectivamente. El objetivo es comparar dos medias poblacionales μ_1 y μ_2. Para ello, lo adecuado es basarse en el estimador $\bar{x}_1 - \bar{x}_2$, diferencia entre ambas medias muestrales.

La situación que se estudia, en esta comparación, es la de dos muestras aleatorias independientes del mismo tamaño y suficientemente grandes, de manera que puede asegurarse la normalidad aproximada de ambas medias muestrales.

Se analizará si existe una diferencia significativa entre el objetivo medio conseguido con el enfriamiento parcial para cada porcentaje del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar. Como las varianzas poblacionales son finitas y desconocidas, las sustituiremos por las cuasivarianzas muestrales S_1^2 y S_2^2, dando lugar al intervalo:

$$\left(-z_{\alpha/2} \leq Z \leq z_{\alpha/2} \right)$$

$$Z = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$$

- Si $Z < -z_{\alpha/2}$, es mejor el enfriamiento con temperatura inicial estándar
- Si $Z > z_{\alpha/2}$, es mejor el enfriamiento parcial
- Si $\left(-z_{\alpha/2} \leq Z \leq z_{\alpha/2} \right)$, la prueba no permite afirmar que existan diferencias significativas entre ambos casos

$\bar{x}_1$ = *media muestral (100%)*

$\bar{x}_2$ = *media muestral (5%, 10%, 20%,..., 90%)*

n_1 = *tamaño muestra (100%)*

n_2 = *tamaño muestra (5%, 10%, 20%,..., 90%)*

S_1 = *cuasivarianza muestral (100%)*

S_2 = *cuasivarianza muestral (5%, 10%, 20%,..., 90%)*

Comparación entre el objetivo medio obtenido con un enfriamiento parcial al 90% del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
90%	Objetivo	6964,83	390,59

$$(-1{,}96 \leq 1{,}37 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
90%	Objetivo	35666,47	1467,09

$$(-1{,}96 \leq -1{,}54 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
90%	Objetivo	26523177,45	46896,42

$$Z = -2{,}03 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
90%	Objetivo	250453507	387819,59

$$Z = 2{,}35 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.6: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 90% del total de ciclos y el enfriamiento con temperatura inicial estándar

- En TSP 47 y TSP 80 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre el algoritmo con enfriamiento parcial al 90% del total de ciclos y el algoritmo con temperatura inicial estándar.
- En QAP 47 se puede considerar que el algoritmo con temperatura inicial estándar proporciona mejores soluciones que el algoritmo con enfriamiento parcial al 90% del total de ciclos.
- En QAP 80 se puede considerar que el algoritmo con enfriamiento parcial al 90% del total de ciclos proporciona mejores soluciones que el algoritmo con temperatura inicial estándar.

Comparación entre el objetivo medio obtenido con un enfriamiento parcial al 80% del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
80%	Objetivo	6970,64	392,03

$$(-1{,}96 \leq 1{,}27 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
80%	Objetivo	35297,91	1592,95

$$(-1{,}96 \leq 0{,}16 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
80%	Objetivo	26520680,57	45713,59

$$(-1{,}96 \leq -0{,}63 \leq 1{,}96)$$

QAP 80:

		MED	DESV
100%	Objetivo	250505167	323685,34

		MED	DESV
80%	Objetivo	250541184,8	325238,13

$$(-1{,}96 \leq 0{,}71 \leq 1{,}96)$$

Tabla 4.7: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 80% del total de ciclos y el enfriamiento con temperatura inicial estándar

- En las dos instancias de los problemas TSP y QAP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre el algoritmo con enfriamiento parcial al 80% del total de ciclos y el algoritmo con temperatura inicial estándar.

Comparación entre el objetivo medio obtenido con un enfriamiento parcial al 70% del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
70%	Objetivo	7063,76	363,37

$$(-1{,}96 \leq -0{,}44 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
70%	Objetivo	35400,44	1659,42

$$(-1{,}96 \leq -0{,}28 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
70%	Objetivo	26518716,55	44233,08

$$(-1{,}96 \leq -1{,}35 \leq 1{,}96)$$

QAP 80:

		MED	DESV
100%	Objetivo	250505167	323685,34

		MED	DESV
70%	Objetivo	250580944,8	329969,12

$$(-1{,}96 \leq -0{,}14 \leq 1{,}96)$$

Tabla 4.8: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 70% del total de ciclos y el enfriamiento con temperatura inicial estándar

- En las dos instancias de los problemas TSP y QAP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre el algoritmo con enfriamiento parcial al 70% del total de ciclos y el algoritmo con temperatura inicial estándar.

Comparación entre el objetivo medio obtenido con un enfriamiento parcial al 60% del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
60%	Objetivo	7005,08	418,25

$$(-1{,}96 \leq 0{,}61 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
60%	Objetivo	35389,53	1707,95

$$(-1{,}96 \leq -0{,}23 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
60%	Objetivo	26509624,83	39781,98

$$(-1{,}96 \leq 0{,}15 \leq 1{,}96)$$

QAP 80:

		MED	DESV
100%	Objetivo	250505167	323685,34

		MED	DESV
60%	Objetivo	250503326,5	351551,79

$$(-1{,}96 \leq 1{,}46 \leq 1{,}96)$$

Tabla 4.9: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 60% del total de ciclos y el enfriamiento con temperatura inicial estándar

- En las dos instancias de los problemas TSP y QAP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre el algoritmo con enfriamiento parcial al 60% del total de ciclos y el algoritmo con temperatura inicial estándar.

Comparación entre el objetivo medio obtenido con un enfriamiento parcial al 50% del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
50%	Objetivo	7045,54	348,42

$$(-1{,}96 \leq -0{,}1 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
50%	Objetivo	35458,71	1618,69

$$(-1{,}96 \leq -0{,}54 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
50%	Objetivo	26507920,5	45163,54

$$(-1{,}96 \leq 0{,}41 \leq 1{,}96)$$

QAP 80:

		MED	DESV
100%	Objetivo	250505167	323685,34

		MED	DESV
50%	Objetivo	250501889,1	333451,58

$$(-1{,}96 \leq 1{,}53 \leq 1{,}96)$$

Tabla 4.10: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 50% del total de ciclos y el enfriamiento con temperatura inicial estándar

- En las dos instancias de los problemas TSP y QAP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre el algoritmo con enfriamiento parcial al 50% del total de ciclos y el algoritmo con temperatura inicial estándar.

Comparación entre el objetivo medio obtenido con un enfriamiento parcial al 40% del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
40%	Objetivo	7040,21	442,63

$$(-1{,}96 \leq -0{,}0006 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
40%	Objetivo	35668,78	1605,24

$$(-1{,}96 \leq -1{,}48 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
40%	Objetivo	26526871,8	44315,1

$$Z = -2{,}7 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250505167	323685,34

		MED	DESV
40%	Objetivo	250658076,3	364640,68

$$(-1{,}96 \leq -1{,}69 \leq 1{,}96)$$

Tabla 4.11: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 40% del total de ciclos y el enfriamiento con temperatura inicial estándar

- En las dos instancias de TSP y en QAP 80 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre el algoritmo con enfriamiento parcial al 40% del total de ciclos y el algoritmo con temperatura inicial estándar.
- En QAP 47 se puede considerar que el algoritmo con temperatura inicial estándar proporciona mejores soluciones que el algoritmo con enfriamiento parcial al 40% del total de ciclos.

Comparación entre el objetivo medio obtenido con un enfriamiento parcial al 30% del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
30%	Objetivo	7103,36	329,96

$$(-1{,}96 \leq -1{,}25 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
30%	Objetivo	37686,68	1804,39

$$Z = -9{,}82 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
30%	Objetivo	26589921,47	73187,13

$$Z = -9{,}44 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250505167	323685,34

		MED	DESV
30%	Objetivo	251518934,4	544294,8

$$Z = -14{,}77 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.12: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 30% del total de ciclos y el enfriamiento con temperatura inicial estándar

- En TSP 47 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre el algoritmo con enfriamiento parcial al 30% del total de ciclos y el algoritmo con temperatura inicial estándar.

- En las dos instancias de QAP y en TSP 80 se puede considerar que el algoritmo con temperatura inicial estándar proporciona mejores soluciones que el algoritmo con enfriamiento parcial al 30% del total de ciclos.

Comparación entre el objetivo medio obtenido con un enfriamiento parcial al 20% del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
20%	Objetivo	7479,11	417,95

$$Z = -7{,}74 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
20%	Objetivo	39339,81	1881,37

$$Z = -16{,}32 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
20%	Objetivo	26629803,4	76495,26

$$Z = -13{,}71 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250505167	323685,34

		MED	DESV
20%	Objetivo	251816859,3	679054,76

$$Z = -16{,}4 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.13: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 20% del total de ciclos y el enfriamiento con temperatura inicial estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo con temperatura inicial estándar proporciona mejores soluciones que el algoritmo con enfriamiento parcial al 20% del total de ciclos.

Comparación entre el objetivo medio obtenido con un enfriamiento parcial al 10% del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
10%	Objetivo	7983,96	510,31

$$Z = -14{,}8 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
10%	Objetivo	40662,46	2528,46

$$Z = -17{,}88 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
10%	Objetivo	26666354,57	82103,48

$$Z = -16{,}94 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250505167	323685,34

		MED	DESV
10%	Objetivo	252323083,9	670236,78

$$Z = -23{,}33 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.14: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 10% del total de ciclos y el enfriamiento con temperatura inicial estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo con temperatura inicial estándar proporciona mejores soluciones que el algoritmo con enfriamiento parcial al 10% del total de ciclos.

Comparación entre el objetivo medio obtenido con un enfriamiento parcial al 5% del total de ciclos y el objetivo medio obtenido con la temperatura inicial estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
5%	Objetivo	8210,16	617,29

$$Z = -16{,}1 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
5%	Objetivo	41710,48	2750,48

$$Z = -20{,}11 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
5%	Objetivo	26698708,9	93314,03

$$Z = -18{,}43 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250505167	323685,34

		MED	DESV
5%	Objetivo	252415960,4	670898,6

$$Z = -24{,}55 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.15: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento parcial al 5% del total de ciclos y el enfriamiento con temperatura inicial estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo con temperatura inicial estándar proporciona mejores soluciones que el algoritmo con enfriamiento parcial al 5% del total de ciclos.

A continuación se recoge en forma resumida la información de la comparativa en una tabla de comparación de objetivos.

En la tabla 4.16, el símbolo '+' significa que el algoritmo con temperatura inicial estándar da mejores soluciones, '=' quiere decir que la diferencia no es significativa, y '-' significa que el algoritmo con temperatura inicial estándar ofrece soluciones de peor calidad.

Instancia	90%	80%	70%	60%	50%	40%	30%	20%	10%	5%
TSP 47	=	=	=	=	=	=	=	+	+	+
TSP 80	=	=	=	=	=	=	+	+	+	+
QAP 47	+	=	=	=	=	+	+	+	+	+
QAP 80	-	=	=	=	=	=	+	+	+	+

Tabla 4.16: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de objetivos según la distribución normal.

Conclusiones

Con respecto al estudio de *la influencia de la* temperatura inicial en el comportamiento del algoritmo, analizando los resultados y gráficas obtenidos, puede observarse que cuanto menor es el porcentaje utilizado en el enfriamiento parcial, menor es el tiempo de ejecución del algoritmo, y como era previsible, cuanto menor es este porcentaje peores son las soluciones obtenidas.

Tras el análisis estadístico realizado, se puede concluir que para un porcentaje de ciclos de enfriamiento igual o superior al 50% la calidad de la solución media obtenida (objetivo) no es significativamente diferente (peor) a la obtenida por el algoritmo con temperatura inicial estándar.

4.2. INFLUENCIA DEL PARÁMETRO α EN LA EFICIENCIA DEL ALGORITMO

El objetivo que se persigue con este estudio es valorar el comportamiento del algoritmo cuando en el proceso de reducción monótona de la temperatura mediante el esquema multiplicativo exponencial, se modifica el factor α dinámicamente tras cada ciclo de enfriamiento.

4.2.1. Elección de parámetros de enfriamiento

Para realizar el estudio, se va a modificar dinámicamente el parámetro α de la ecuación de reducción de la temperatura del enfriamiento multiplicativo exponencial, con valores iniciales más bajos, α = 0,8, y valores finales cercanos a 1, α = 0,99, de manera que se pueda establecer una comparación con el caso estándar, donde el parámetro es constante con valor α = 0,95, tanto en la calidad de las soluciones como en el número de iteraciones efectuadas.

El resto de los parámetros de enfriamiento, temperatura inicial T_0 y número L de transiciones de estado para cada valor de la temperatura, se mantienen como en el enfriamiento multiplicativo exponencial estándar estudiado en el capítulo 3.

Se utilizan 6 porcentajes de variación dinámica: 1%, 2%, 3%, 4%, 5% y 10%.

La fórmula de variación del parámetro α es:

$$\alpha_k = \begin{cases} 0{,}8 & \text{si } k = 0 \\ \alpha_{k-1} + (0{,}99 - \alpha_{k-1})p & \text{si } k > 0 \end{cases}$$

donde:

- k es el ciclo de enfriamiento.
- p es el tanto por uno de incremento del factor α, siendo $p = 0'01(1\%), 0'02(2\%), 0'03(3\%), 0'04(4\%), 0'05(5\%), 0'1(10\%)$.

El porcentaje de incremento *p* del factor α determina directamente la velocidad del enfriamiento en el algoritmo del recocido simulado y, por tanto, el tiempo de proceso. Un porcentaje de incremento pequeño determina valores de α próximos a 0,8 durante buena parte de la ejecución, produciendo un descenso rápido de la temperatura y con ello una convergencia rápida del algoritmo, mientras que un porcentaje de incremento grande determina valores de α próximos a 0,99 ya en los ciclos iniciales de enfriamiento, conllevando un descenso lento de la temperatura y una convergencia más lenta.

4.2.2. Resultados de la pruebas

Las tablas 4.17 – 4.20, figuras 4.9 – 4.16 muestran los resultados de las pruebas para los diferentes porcentajes de variación dinámica del parámetro α utilizados en cada instancia de los problemas TSP y QAP.

Para cada instancia de los problemas se han realizado 100 ejecuciones con el programa de enfriamiento exponencial, obteniéndose el valor medio y la desviación típica tanto del objetivo logrado como del número total de iteraciones empleadas en cada caso.

		MED	**DESV**
variación dinámica α = *1%*	Objetivo	7193,99	347,58
	Iteraciones	152751,09	14250,41
variación dinámica α = *2%*	Objetivo	7090,46	289,68
	Iteraciones	209237,65	26082,06
variación dinámica α = *3%*	Objetivo	7078,23	436,71
	Iteraciones	346416,85	56163,80
variación dinámica α = *4%*	Objetivo	7020,52	401,15
	Iteraciones	764352,82	82349,53
variación dinámica α = *5%*	Objetivo	7039,88	387,99
	Iteraciones	1117424,93	82569,11
variación dinámica α = *10%*	Objetivo	7070,77	395,40
	Iteraciones	1785367,91	91763,52
valor constante α = *0,95%*	Objetivo	7040,17	383,06
	Iteraciones	498561,6	28723,43

Tabla 4.17: Resultados TSP 47

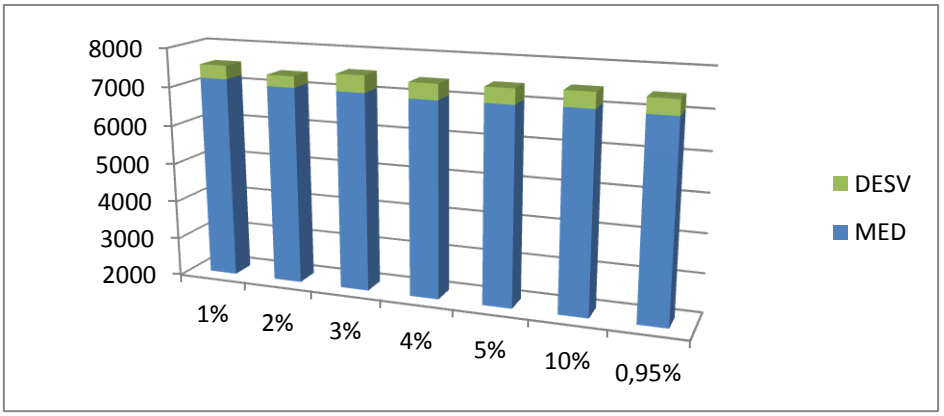

Figura 4.9: Comparación de resultados (objetivo) TSP 47

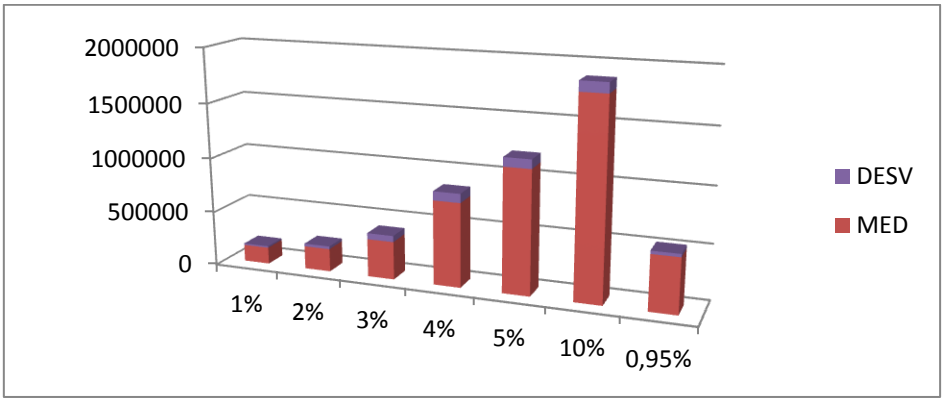

Figura 4.10: Comparación de resultados (nº iteraciones) TSP 47

		MED	DESV
variación dinámica α = *1%*	Objetivo	35582,09	1403,75
	Iteraciones	574305,57	53272,15
variación dinámica α = *2%*	Objetivo	35392,03	1566,66
	Iteraciones	941989,48	174809,01
variación dinámica α = *3%*	Objetivo	35930,44	1487,95
	Iteraciones	2252960,9	307245,52
variación dinámica α = *4%*	Objetivo	36134,55	1803,87
	Iteraciones	3933665,66	372791,11
variación dinámica α = *5%*	Objetivo	36202,86	1754,11
	Iteraciones	4906738,29	317405,4
variación dinámica α = *10%*	Objetivo	35608,25	1536,68
	Iteraciones	6856844,13	378071,09
valor constante α = *0,95%*	Objetivo	35334,72	1574,20
	Iteraciones	1761604,4	93123,83

Tabla 4.18: Resultados TSP 80

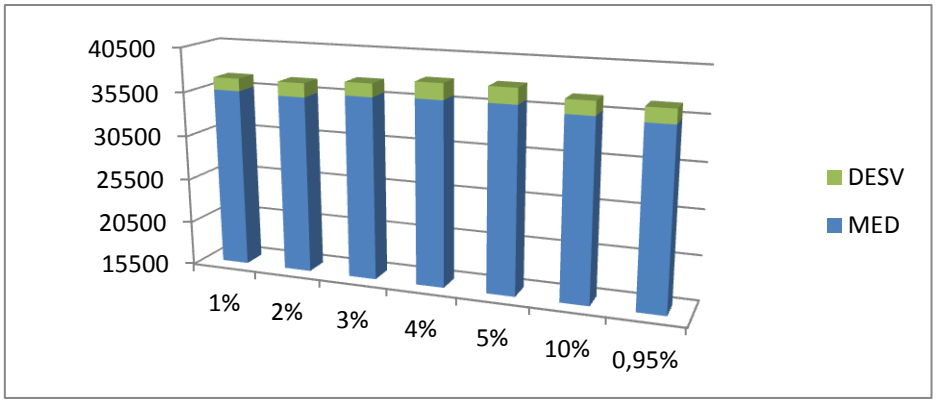

Figura 4.11: Comparación de resultados (objetivo) TSP 80

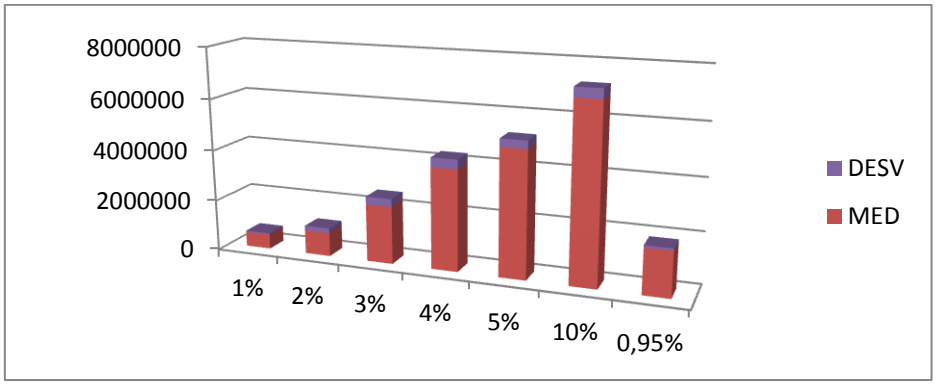

Figura 4.12: Comparación de resultados (nº iteraciones) TSP 80

		MED	DESV
variación dinámica α = *1%*	Objetivo	26524996,44	43825,97
	Iteraciones	182271,37	19402,15
variación dinámica α = *2%*	Objetivo	26523721,21	41960,44
	Iteraciones	254271,04	35192,94
variación dinámica α = *3%*	Objetivo	26520172,12	45557,83
	Iteraciones	552057,94	105373,46
variación dinámica α = *4%*	Objetivo	26515980,48	46277,48
	Iteraciones	1047471,7	104567,20
variación dinámica α = *5%*	Objetivo	26508269,85	40597,08
	Iteraciones	1382269,97	106937,92
variación dinámica α = *10%*	Objetivo	26502581,24	37814,92
	Iteraciones	2070574,69	132367,76
valor constante α = *0,95%*	Objetivo	26510487,17	41489,66
	Iteraciones	563650,78	38893,80

Tabla 4.19: Resultados QAP 47

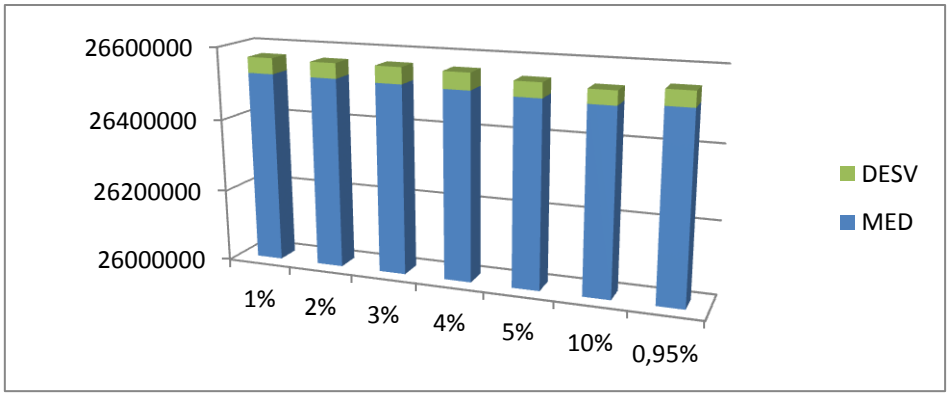

Figura 4.13: Comparación de resultados (objetivo) QAP 47

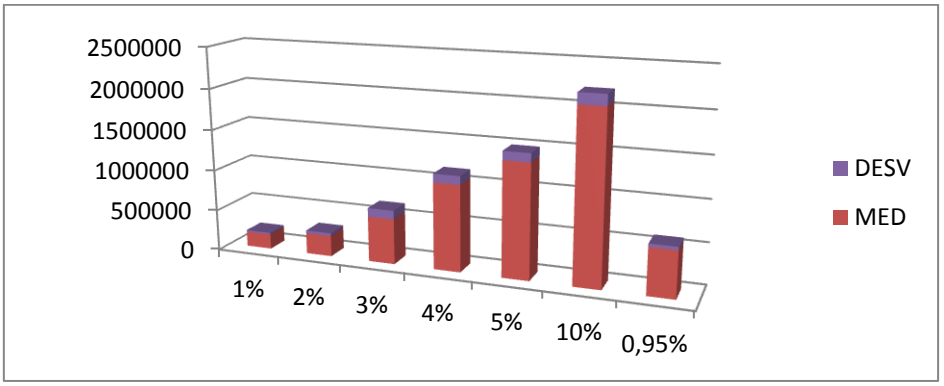

Figura 4.14: Comparación de resultados (nº iteraciones) QAP 47

		MED	DESV
variación dinámica α = *1%*	Objetivo	250715696,7	331074,26
	Iteraciones	536855,97	53289,83
variación dinámica α = *2%*	Objetivo	250530965,2	325964,64
	Iteraciones	805650,77	104523,56
variación dinámica α = *3%*	Objetivo	250471603,8	304342,42
	Iteraciones	1718351,52	291503,93
variación dinámica α = *4%*	Objetivo	250461166,6	320217,87
	Iteraciones	3235493,56	327642,11
variación dinámica α = *5%*	Objetivo	250421203,3	336469,57
	Iteraciones	4233088,6	305010,60
variación dinámica α = *10%*	Objetivo	250455210,9	309412,84
	Iteraciones	6183963,78	329347,44
valor constante α = *0,95%*	Objetivo	250574322,1	335723,84
	Iteraciones	1663308,76	100600,05

Tabla 4.20: Resultados QAP 80

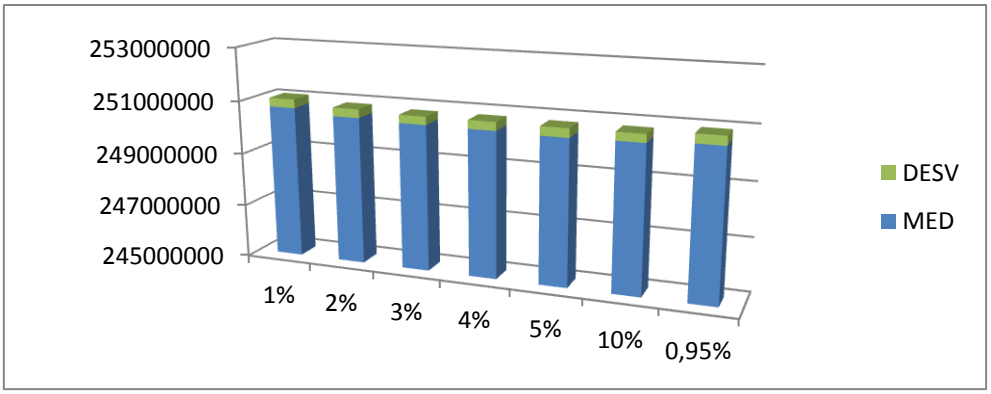

Figura 4.15: Comparación de resultados (objetivo) QAP 80

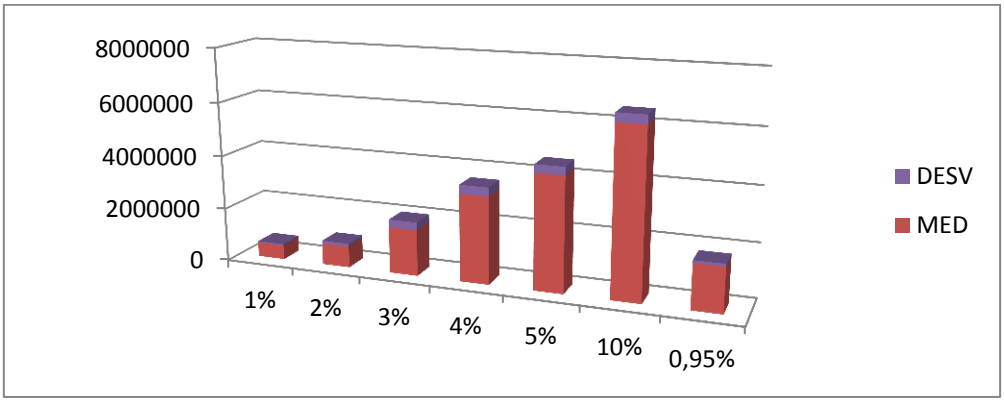

Figura 4.16: Comparación de resultados (nº iteraciones) QAP 80

4.2.3. Análisis de los resultados

A continuación, se va a evaluar el comportamiento del algoritmo en los diferentes escenarios planteados – cuando la variación dinámica de parámetro α se produce desde el valor 0,8 al valor 0,99 con diferentes porcentajes de incremento del 1%, 2%, 3%, 4%, 5% y 10% –.

Prueba estadística

En el estudio se van a realizar sendas estimaciones estadísticas por intervalo al 95% de confianza, para la diferencia de objetivos medios y la diferencia de iteraciones medias respectivamente, tomando como referencia el objetivo medio y el número medio de iteraciones obtenidos con el enfriamiento correspondiente a la temperatura inicial estándar. Con ello se debe verificar:

- Con qué porcentaje (1%, 2%, 3%, 4%, 5% y 10%) el objetivo medio no es significativamente diferente (mejor/peor) que el obtenido con el parámetro constante α = 0,95.

- Con qué porcentaje (1%, 2%, 3%, 4%, 5% y 10%) el número medio de iteraciones no es significativamente distinto (menor/mayor) que el obtenido con el parámetro constante α = 0,95.

Para analizar si puede inferirse una diferencia significativa se va a realizar una estimación por intervalo de confianza para la diferencia de medias, tanto de objetivo como del número de iteraciones. En cada caso se dispone de dos muestras aleatorias independientes de tamaño n_1 y n_2 respectivamente. El objetivo es comparar dos medias poblacionales μ_1 y μ_2. Para ello, lo adecuado es basarse en el estimador $\bar{x}_1 - \bar{x}_2$, diferencia entre ambas medias muestrales.

La situación que se estudia, en esta comparación, es la de dos muestras aleatorias independientes del mismo tamaño y suficientemente grandes, de manera que puede asegurarse la normalidad aproximada de ambas medias muestrales.

Se analizará si existe una diferencia significativa entre el objetivo medio conseguido con el enfriamiento para cada porcentaje de variación dinámica del parámetro α y el objetivo medio obtenido con parámetro constante α = 0,95. Igualmente se procederá en el estudio del número medio de iteraciones. Como las varianzas poblacionales son finitas y desconocidas, las sustituiremos por las cuasivarianzas muestrales S_1^2 y S_2^2, dando lugar al intervalo:

$$\left(-z_{\alpha/2} \leq Z \leq z_{\alpha/2} \right)$$

$$Z = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$$

- Si $Z < -z_{\alpha/2}$, es mejor el enfriamiento con parámetro α constante

- Si $Z > z_{\alpha/2}$, es mejor el enfriamiento con parámetro α dinámico
- Si $\left(-z_{\alpha/2} \leq Z \leq z_{\alpha/2}\right)$, la prueba no permite afirmar que existan diferencias significativas entre ambos casos

$\bar{x}_1$ = *media muestral (100%)*

$\bar{x}_2$ = *media muestral (1%, 2%, 3%, 4%, 5% y 10%)*

n_1 = *tamaño muestra (100%)*

n_2 = *tamaño muestra (1%, 2%, 3%, 4%, 5% y 10%)*

S_1 = *cuasivarianza muestral (100%)*

S_2 = *cuasivarianza muestral (1%, 2%, 3%, 4%, 5% y 10%)*

Comparación entre el enfriamiento con variación dinámica del parámetro α del 1% y el enfriamiento con parámetro α constante.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
1%	Objetivo	7193,99	347,58

$$Z = -2{,}97 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
1%	Objetivo	35582,09	1403,75

$$(-1{,}96 \leq -1{,}17 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
1%	Objetivo	26524996,44	43825,97

$$Z = -2{,}4 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
1%	Objetivo	250715696,7	331074,26

$$Z = -2{,}99 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.21: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 1% y el enfriamiento con parámetro α constante

- En TSP 47 y en las dos instancias del problema QAP se puede considerar que el enfriamiento con parámetro α constante proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 1%.
- En TSP 80 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
1%	Iteraciones	152751,09	14250,41

$Z = 107{,}85 > 1{,}96 = z_{\alpha/2}$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
1%	Iteraciones	574305,57	53272,15

$Z = 110{,}67 > 1{,}96 = z_{\alpha/2}$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
1%	Iteraciones	182271,37	19402,15

$Z = 87{,}74 > 1{,}96 = z_{\alpha/2}$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
1%	Iteraciones	536855,97	53289,83

$Z = 98{,}95 > 1{,}96 = z_{\alpha/2}$

Tabla 4.22: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 1% y el enfriamiento con parámetro α constante

- En las dos instancias de los problemas TSP y QAP se puede considerar que el enfriamiento con variación dinámica del parámetro α del 1% converge más rápidamente que el enfriamiento con parámetro α constante.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 2% y el enfriamiento con parámetro α constante.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
2%	Objetivo	7090,46	289,68

$$(-1{,}96 \leq -1{,}04 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
2%	Objetivo	35392,03	1566,66

$$(-1{,}96 \leq -0{,}26 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
2%	Objetivo	26523721,21	41960,44

$$Z = -2{,}24 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
2%	Objetivo	250530965,2	325964,64

$$(-1{,}96 \leq 0{,}92 \leq 1{,}96)$$

Tabla 4.23: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 2% y el enfriamiento con parámetro α constante

- En las dos instancias del problema TSP y en QAP 80, la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.
- En TSP 47 se puede considerar que el enfriamiento con parámetro α constante proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 2%.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
2%	Iteraciones	209237,65	26082,06

$$Z = 70{,}57 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
2%	Iteraciones	941989,48	174809,01

$$Z = 41{,}38 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
2%	Iteraciones	254271,04	35192,94

$$Z = 58{,}98 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
2%	Iteraciones	805650,77	104523,56

$$Z = 59{,}12 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.24: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 2% y el enfriamiento con parámetro α constante

- En las dos instancias de los problemas TSP y QAP se puede considerar que el enfriamiento con variación dinámica del parámetro α del 2% converge más rápidamente que el enfriamiento con parámetro α constante.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 3% y el enfriamiento con parámetro α constante.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
3%	Objetivo	7078,23	436,71

$$(-1{,}96 \leq -0{,}65 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
3%	Objetivo	35930,44	1487,95

$$Z = -2{,}75 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
3%	Objetivo	26523721,21	41960,44

$$Z = -2{,}24 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
3%	Objetivo	250471603,8	304342,42

$$Z = 2{,}26 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.25: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 3% y el enfriamiento con parámetro α constante

- En TSP 47 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.
- En TSP 80 y en QAP 47, se puede considerar que el enfriamiento con parámetro α constante proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 3%.
- En QAP 80 se puede considerar que el enfriamiento con variación dinámica del parámetro α del 3% proporciona mejores soluciones que el enfriamiento con parámetro α constante.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
3%	Iteraciones	346416,85	56163,8

$$Z = 24{,}12 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
3%	Iteraciones	2252960,9	307245,52

$$Z = -15{,}3 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
3%	Iteraciones	552057,94	105373,46

$$(-1{,}96 \leq 1{,}03 \leq 1{,}96)$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
3%	Iteraciones	1718351,52	291503,93

$$(-1{,}96 \leq -1{,}78 \leq 1{,}96)$$

Tabla 4.26: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 3% y el enfriamiento con parámetro α constante

- En TSP 47 se puede considerar que el enfriamiento con variación dinámica del parámetro α del 3% converge más rápidamente que el enfriamiento con parámetro α constante.
- En TSP 80 se puede considerar que el enfriamiento con parámetro α constante converge más rápidamente que el enfriamiento con variación dinámica del parámetro α del 3%.
- En las dos instancias del QAP, la prueba no permite afirmar que existan diferencias significativas en velocidad de convergencia entre ambos casos.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 4% y el enfriamiento con parámetro α constante.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
4%	Objetivo	7020,52	401,15

$$(-1{,}96 \leq 0{,}35 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
4%	Objetivo	36134,55	1803,87

$$Z = -3{,}34 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
4%	Objetivo	26515980,48	46277,48

$$(-1{,}96 \leq -0{,}88 \leq 1{,}96)$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
4%	Objetivo	250461166,6	320217,87

$$Z = 2{,}44 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.27: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 4% y el enfriamiento con parámetro α constante

- En TSP 47 y en QAP 47, la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.
- En TSP 80 se puede considerar que el enfriamiento con parámetro α constante proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 4%.
- En QAP 80 se puede considerar que el enfriamiento con variación dinámica del parámetro α del 4% proporciona mejores soluciones que el enfriamiento con parámetro α constante.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
4%	Iteraciones	764352,82	82349,53

$$Z = -30{,}47 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	35334,72	1574,21

		MED	DESV
4%	Iteraciones	3933665,66	372791,11

$$Z = -56{,}53 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
4%	Iteraciones	1047471,7	104567,2

$$Z = -43{,}36 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
4%	Iteraciones	3235493,56	327642,11

$$Z = -45{,}87 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.28: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 4% y el enfriamiento con parámetro α constante

- En las dos instancias de los problemas TSP y QAP, se puede considerar que el enfriamiento con parámetro α constante converge más rápidamente que el enfriamiento con variación dinámica del parámetro α del 4%.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 5% y el enfriamiento con parámetro α constante.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
5%	Objetivo	7039,88	387,99

$$(-1{,}96 \leq 0{,}005 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
5%	Objetivo	36202,86	1754,11

$$Z = -3{,}68 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
5%	Objetivo	26508269,85	40597,08

$$(-1{,}96 \leq 0{,}38 \leq 1{,}96)$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
5%	Objetivo	250421203,3	336469,57

$$Z = 3{,}22 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.29: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 5% y el enfriamiento con parámetro α constante

- En TSP 47 y en QAP 47, la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.
- En TSP 80 se puede considerar que el enfriamiento con parámetro α constante proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 5%.
- En QAP 80 se puede considerar que el enfriamiento con variación dinámica del parámetro α del 5% proporciona mejores soluciones que el enfriamiento con parámetro α constante.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
5%	Iteraciones	1117424,93	82569,11

$$Z = -70{,}79 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
5%	Iteraciones	4906738,29	317405,4

$$Z = -95{,}08 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
5%	Iteraciones	1382269,97	106937,92

$$Z = -43{,}36 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
5%	Iteraciones	4233088,6	305010,6

$$Z = -80{,}01 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.30: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 5% y el enfriamiento con parámetro α constante

- En las dos instancias de los problemas TSP y QAP, se puede considerar que el enfriamiento con parámetro α constante converge más rápidamente que el enfriamiento con variación dinámica del parámetro α del 5%.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 10% y el enfriamiento con parámetro α constante.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
10%	Objetivo	7070,77	395,4

$$(-1{,}96 \leq -0{,}55 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
10%	Objetivo	35608,25	1536,68

$$(-1{,}96 \leq -1{,}24 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
10%	Objetivo	26502581,24	37814,92

$$(-1{,}96 \leq 1{,}4 \leq 1{,}96)$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
10%	Objetivo	250455210,9	309412,84

$$Z = 2{,}6 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.31: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento con variación dinámica del parámetro α del 10% y el enfriamiento con parámetro α constante

- En las dos instancias del problema TSP y en QAP 47, la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.
- En QAP 80 se puede considerar que el enfriamiento con variación dinámica del parámetro α del 10% proporciona mejores soluciones que el enfriamiento con parámetro α constante.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
10%	Iteraciones	1785367,91	91763,52

$$Z = -133{,}83 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
10%	Iteraciones	6856844,13	378071,09

$$Z = -130{,}85 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
10%	Iteraciones	2070574,69	132367,76

$$Z = -109{,}27 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
10%	Iteraciones	6183963,78	329347,44

$$Z = -131{,}27 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.32: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento con variación dinámica del parámetro α del 10% y el enfriamiento con parámetro α constante

- En las dos instancias de los problemas TSP y QAP, se puede considerar que el enfriamiento con parámetro α constante converge más rápidamente que el enfriamiento con variación dinámica del parámetro α del 10%.

A continuación se recoge en forma resumida la información de la comparativa en sendas tablas, una sobre objetivos y otra sobre convergencia temporal.

En la tabla 4.33, el símbolo '+' significa que el algoritmo con enfriamiento de parámetro constante α = 0,95 da mejores soluciones, '=' quiere decir que la diferencia no es significativa, y '-' significa que el algoritmo con enfriamiento de parámetro constante α = 0,95 ofrece soluciones de peor calidad.

Instancia	1%	2%	3%	4%	5%	10%
TSP 47	+	=	=	=	=	=
TSP 80	=	=	+	+	+	=
QAP 47	+	+	+	=	=	=
QAP 80	+	=	-	-	-	-

Tabla 4.33: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de objetivos según la distribución normal

En la tabla 4.34, '+' significa que el algoritmo con enfriamiento de parámetro constante α = 0,95 converge más rápido, '=' quiere decir que la diferencia no es significativa, y '-' significa que el algoritmo con enfriamiento de parámetro constante α = 0,95 converge más lentamente.

Instancia	1%	2%	3%	4%	5%	10%
TSP 47	-	-	-	+	+	+
TSP 80	-	-	+	+	+	+
QAP 47	-	-	=	+	+	-
QAP 80	-	-	=	+	+	-

Tabla 4.34: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de número de iteraciones según la distribución normal

Conclusiones

Con respecto al estudio de *la influencia del parámetro* α *en la eficiencia del* algoritmo, analizando los resultados y gráficas obtenidos, puede observarse que cuanto mayor es el porcentaje de incremento de α, mayor es el tiempo de ejecución del algoritmo, ya que la temperatura desciende más lentamente.

Tras el análisis estadístico realizado, se puede concluir que cuando el enfriamiento se produce con un mayor porcentaje de incremento del parámetro α, el objetivo conseguido no es significativamente diferente (mejor) que el obtenido con el enfriamiento de parámetro α constante. Hay un gran aumento en cuanto al tiempo de ejecución pero no hay mejoras del objetivo.

Con respecto al tiempo de ejecución, en las dos instancias de los problemas TSP y QAP se puede observar que el enfriamiento con variación dinámica del parámetro α del 1% o 2% converge más rápidamente que el enfriamiento con parámetro constante α = 0,95. El caso opuesto se presenta cuando el porcentaje de variación dinámica de α es 4%, 5% o 10%, en cuyo caso el tiempo de ejecución aumenta considerablemente. Cuando el porcentaje de variación dinámica de α es del 3%, el tiempo de ejecución es similar al del caso estándar.

También puede observarse que la desviación típica del objetivo obtenida en el caso de variación dinámica del parámetro α del 2% es mejor que la del caso estándar en las dos instancias del problema TSP y es similar en las dos instancias del problema QAP. Por tanto, puede afirmarse que la variante dinámica con porcentaje del 2% constituye un algoritmo más robusto que el del enfriamiento estándar.

La principal conclusión que puede extraerse de la comparativa realizada es que el algoritmo con enfriamiento basado en la variación dinámica del parámetro α del 2% proporciona un objetivo estadísticamente similar al del enfriamiento estándar con una mayor reducción del tiempo de ejecución (iteraciones).

4.3. INFLUENCIA COMBINADA DE LA TEMPERATURA INICIAL Y EL PARÁMETRO α EN LA EFICIENCIA DEL ALGORITMO

Para valorar la eficiencia del algoritmo en la calidad de las soluciones obtenidas respecto al tiempo de ejecución, se va a proceder a un tercer estudio en donde se combinan los dos planteamientos analizados en los apartados 4.1 y 4.2.

4.3.1. Elección de parámetros de enfriamiento

El estudio va a consistir en analizar la influencia de la temperatura inicial y el parámetro α en la eficiencia del algoritmo. En este caso y teniendo en cuenta el análisis de resultados realizado por separado para cada parámetro en los apartados anteriores, se ha decidido considerar los siguientes casos:

- La temperatura inicial T_0 tomará valores relativos a porcentajes del 50%, 60%, 70% y 80% del número de ciclos de enfriamiento correspondientes a la temperatura inicial estándar.
- El parámetro α de reducción de la temperatura variará con porcentajes de incremento del 2%, 3%, y 4%.

La fórmula de cálculo de la temperatura inicial en función del porcentaje de ciclos de enfriamiento a realizar y la fórmula de variación del parámetro α en función del porcentaje de incremento son las mismas que se han utilizado en los estudios de los apartados 4.1 y 4.2. Asimismo, el número L de transiciones de estado para cada valor de la temperatura se mantiene como en el enfriamiento multiplicativo exponencial estándar estudiado en el capítulo 3.

4.3.2. Resultados de la pruebas

Las tablas 4.35 – 4.38, figuras 4.17 – 4.24 muestran los resultados de las pruebas para los diferentes casos combinados del porcentaje de número de ciclos de enfriamiento y del porcentaje de incremento dinámico del parámetro α utilizados en cada instancia de los problemas TSP y QAP.

Para cada instancia de los problemas se han realizado 100 ejecuciones con el programa de enfriamiento exponencial, obteniéndose el valor medio y la desviación típica tanto del objetivo logrado como del número total de iteraciones empleadas en cada caso.

α	T_0		MED	DESV
2%	50%	Objetivo	7160,86	329,58
		Iteraciones	85263,6	14869,27
	60%	Objetivo	7182,15	386,27
		Iteraciones	103296,05	14083,15
	70%	Objetivo	7101,99	352,66
		Iteraciones	123405,22	13866,41
	80%	Objetivo	7116,13	347,319
		Iteraciones	144518,15	15252,51
3%	50%	Objetivo	7082,8	363,31
		Iteraciones	94147,07	17670,41
	60%	Objetivo	7076,37	338,31
		Iteraciones	120736,86	20801,18
	70%	Objetivo	7045,66	293,32
		Iteraciones	155209,37	23034,48
	80%	Objetivo	7087,48	352,33
		Iteraciones	202519,2	31268,65
4%	50%	Objetivo	7113,98	359,79
		Iteraciones	112386,92	21081,71
	60%	Objetivo	7129,09	354,93
		Iteraciones	156160,98	28824,83
	70%	Objetivo	7109,48	396,59
		Iteraciones	226892,92	50520,61
	80%	Objetivo	7068,74	384,24
		Iteraciones	343455,28	62687,19
0.95%	100%	Objetivo	7040,17	383,06
		Iteraciones	498561,6	28723,43

Tabla 4.35: Resultados TSP 47 con los diferentes valores del parámetro α

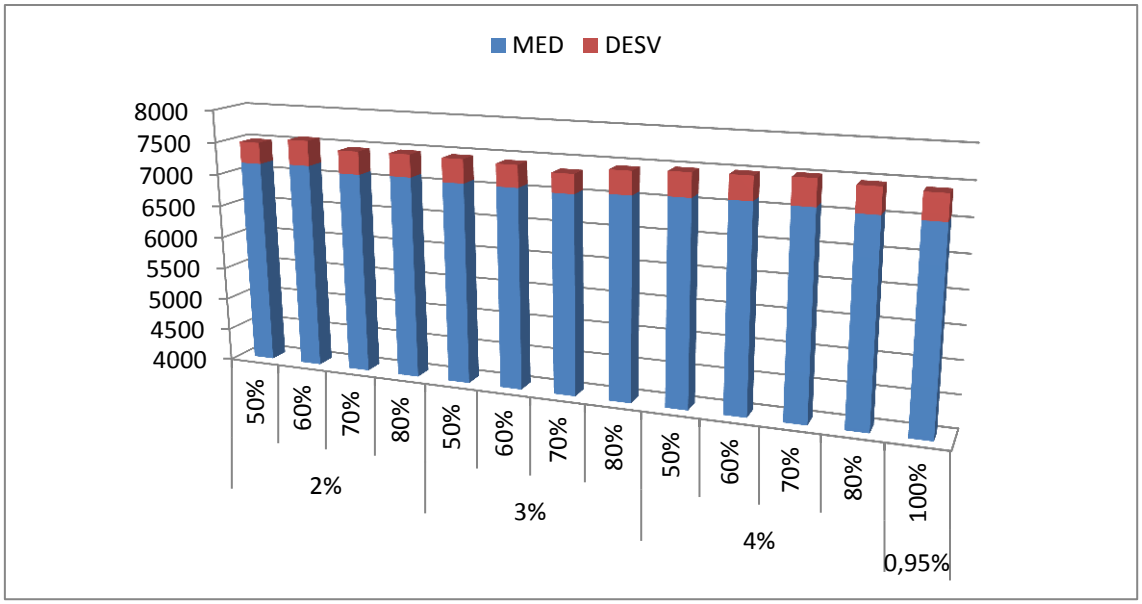

Figura 4.17: Comparación de resultados (objetivo) TSP 47

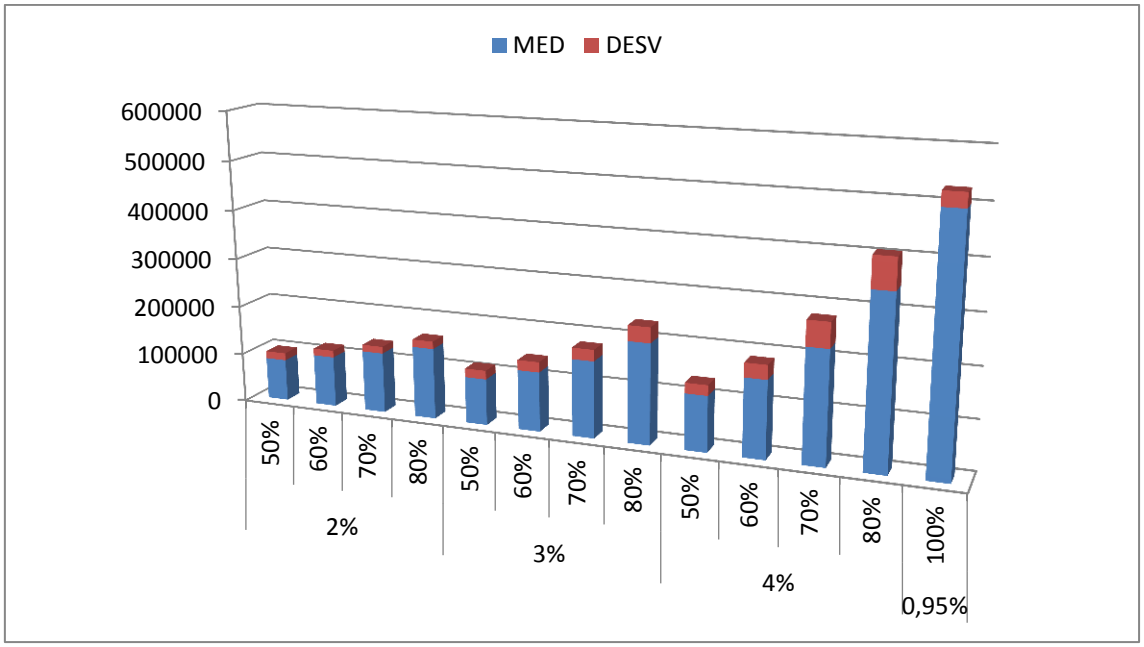

Figura 4.18: Comparación de resultados (nº iteraciones) TSP 47

α	T_0		MED	DESV
2%	50%	Objetivo	35322,18	1548,06
		Iteraciones	317300,31	60728,38
	60%	Objetivo	35609,04	1432,79
		Iteraciones	407841,97	82278,72
	70%	Objetivo	35381,77	1507,14
		Iteraciones	492320,94	75096,85
	80%	Objetivo	35254,95	1567,55
		Iteraciones	601318,09	92365,27
3%	50%	Objetivo	35466,55	1516,0
		Iteraciones	386736,41	91062,19
	60%	Objetivo	35250,82	1492,81
		Iteraciones	504103,8	101378,57
	70%	Objetivo	35569,71	1539,51
		Iteraciones	686269,19	117545,95
	80%	Objetivo	35883,19	1670,77
		Iteraciones	984062,77	236338,04
4%	50%	Objetivo	35900,67	1668,19
		Iteraciones	533943,92	183432,23
	60%	Objetivo	35507,61	1646,57
		Iteraciones	800911,16	232461,12
	70%	Objetivo	35805,83	1527,68
		Iteraciones	1343796,5	339716,66
	80%	Objetivo	35996,34	1803,34
		Iteraciones	2142448,5	297859,62
0,95%	100%	Objetivo	35334,72	1574,20
		Iteraciones	1761604,4	93123,83

Tabla 4.36: Resultados TSP 80 con los diferentes valores del parámetro α

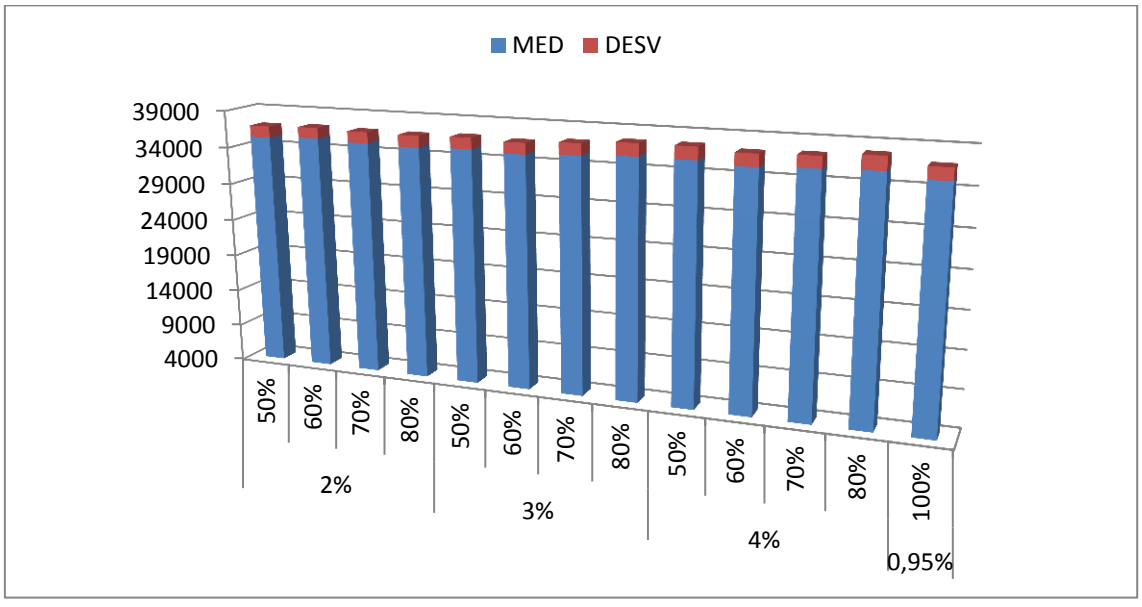

Figura 4.19: Comparación de resultados (objetivo) TSP 80

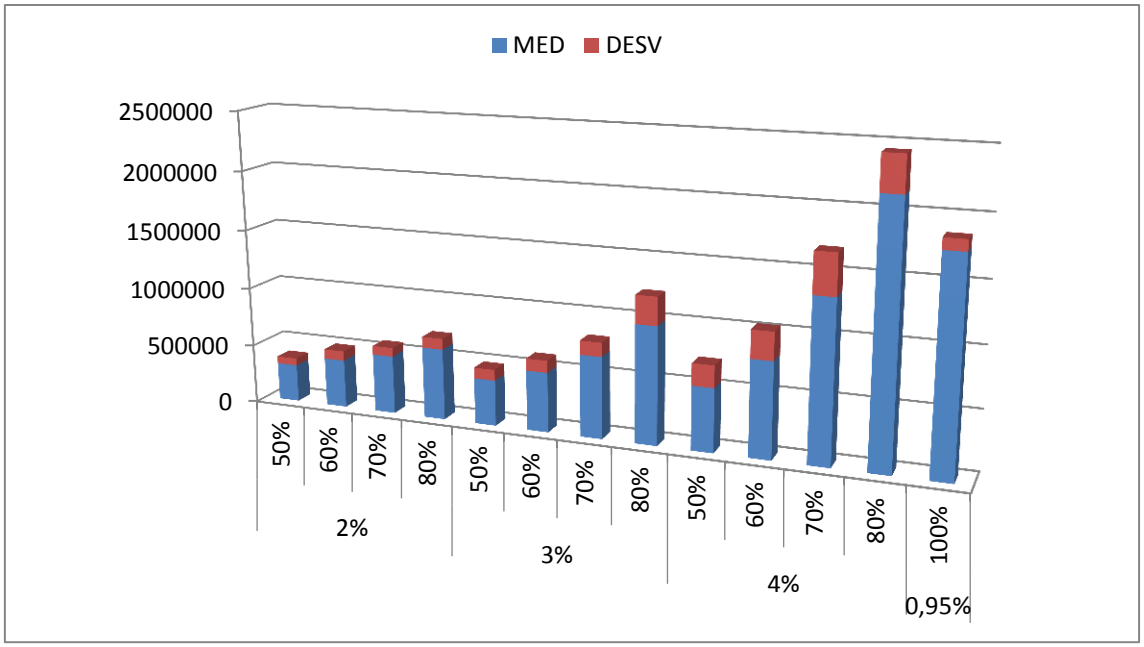

Figura 4.20: Comparación de resultados (nº iteraciones) TSP 80

α	T_0		MED	DESV
2%	**50%**	**Objetivo**	26533714	54019,71
		Iteraciones	100409,47	19488,76
	60%	**Objetivo**	26536665	52746,71
		Iteraciones	123206,16	22438,56
	70%	**Objetivo**	26527046	46819,24
		Iteraciones	149331,18	26519,91
	80%	**Objetivo**	26530536	46846,33
		Iteraciones	182145,71	25363,06
3%	**50%**	**Objetivo**	26537054	48008,02
		Iteraciones	123200,96	28136,52
	60%	**Objetivo**	26520634	48570,62
		Iteraciones	154990,16	35562,9
	70%	**Objetivo**	26527358	49654,78
		Iteraciones	201693,74	46989,12
	80%	**Objetivo**	26526850	47194,62
		Iteraciones	267612,43	60211,27
4%	**50%**	**Objetivo**	26535772	51233,42
		Iteraciones	143296,17	41657,77
	60%	**Objetivo**	26521725	51988,64
		Iteraciones	205491,96	55262,08
	70%	**Objetivo**	26524852	48578,17
		Iteraciones	303450,15	71597,87
	80%	**Objetivo**	26524470	52923,73
		Iteraciones	518908,9	109670,15
0,95%	**100%**	**Objetivo**	26510487,17	41489,66
		Iteraciones	563650,78	38893,80

Tabla 4.37: Resultados QAP 47 con los diferentes valores del parámetro α

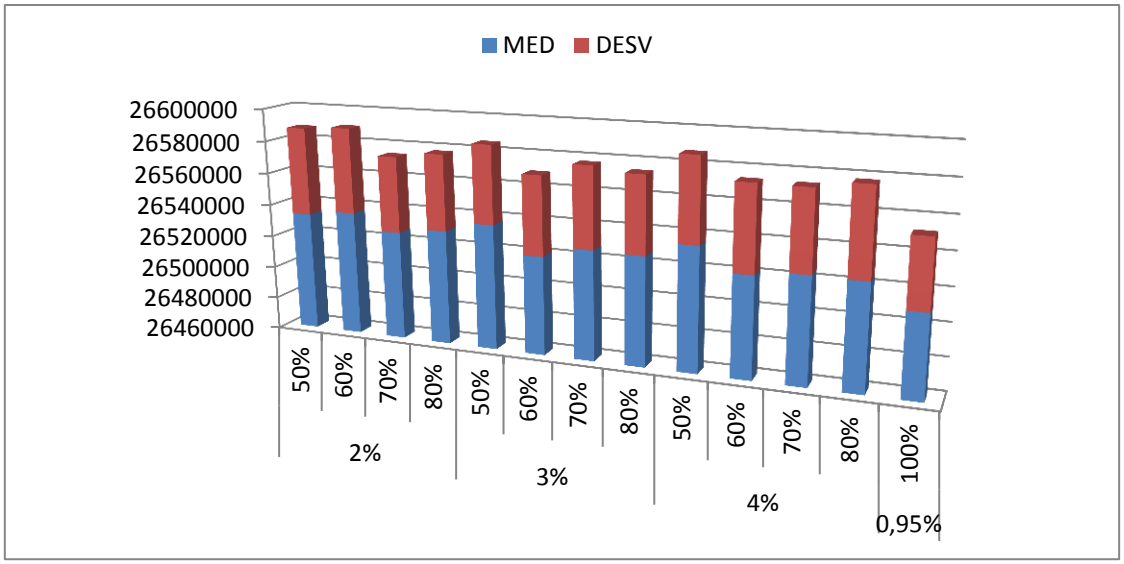

Figura 4.21: Comparación de resultados (objetivo) QAP 47

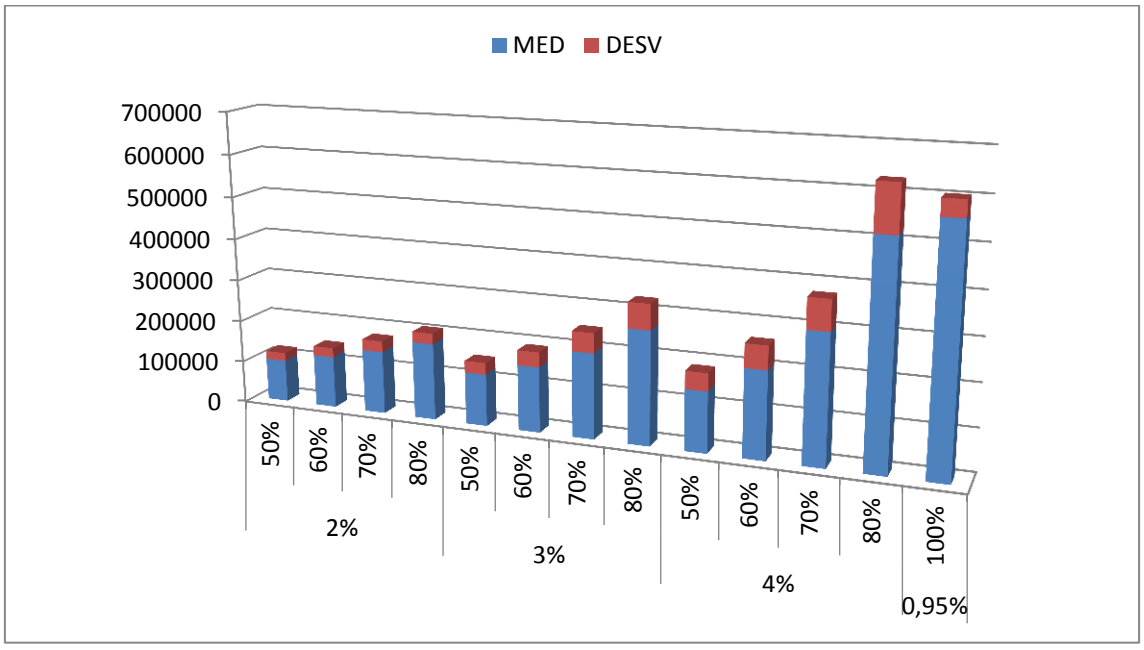

Figura 4.22: Comparación de resultados (nº iteraciones) QAP 47

α	T_0		MED	DESV
2%	**50%**	**Objetivo**	250771702	333069,44
		Iteraciones	293557,53	48569,93
	60%	**Objetivo**	250700410	342584,86
		Iteraciones	432107,22	42287,57
	70%	**Objetivo**	250690324	356043,96
		Iteraciones	444331,99	58282,20
	80%	**Objetivo**	250676638	372644,25
		Iteraciones	542629,24	65429,78
3%	**50%**	**Objetivo**	250750833	390452,76
		Iteraciones	349686,11	76558,72
	60%	**Objetivo**	250604631	351909,83
		Iteraciones	454004,4	89624,66
	70%	**Objetivo**	250661039	379075,09
		Iteraciones	592162,09	112452,62
	80%	**Objetivo**	250639704	387407,99
		Iteraciones	808311,64	150395,21
4%	**50%**	**Objetivo**	250673195	358630,69
		Iteraciones	433985,18	103053,04
	60%	**Objetivo**	250655350	377946,99
		Iteraciones	624125,01	135797,41
	70%	**Objetivo**	250596614	330386,1
		Iteraciones	995925,57	223978,59
	80%	**Objetivo**	250519437	274385,22
		Iteraciones	1599538,5	285546,93
0,95%	**100%**	**Objetivo**	250574322,1	335723,84
		Iteraciones	1663308,76	100600,05

Tabla 4.38: Resultados QAP 80 con los diferentes valores del parámetro α

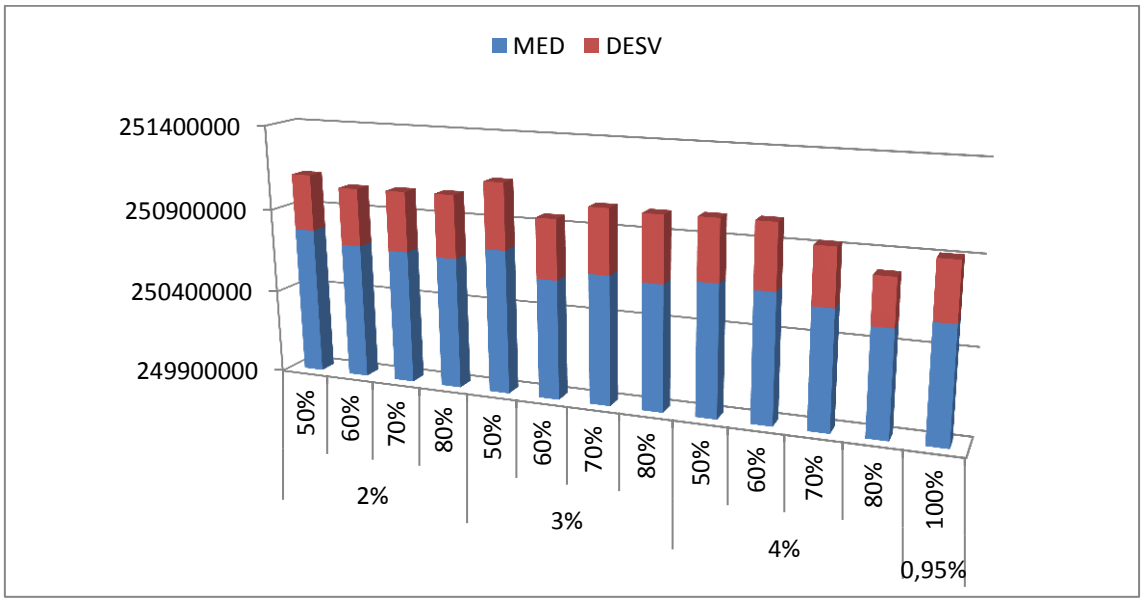

Figura 4.23: Comparación de resultados (objetivo) QAP 80

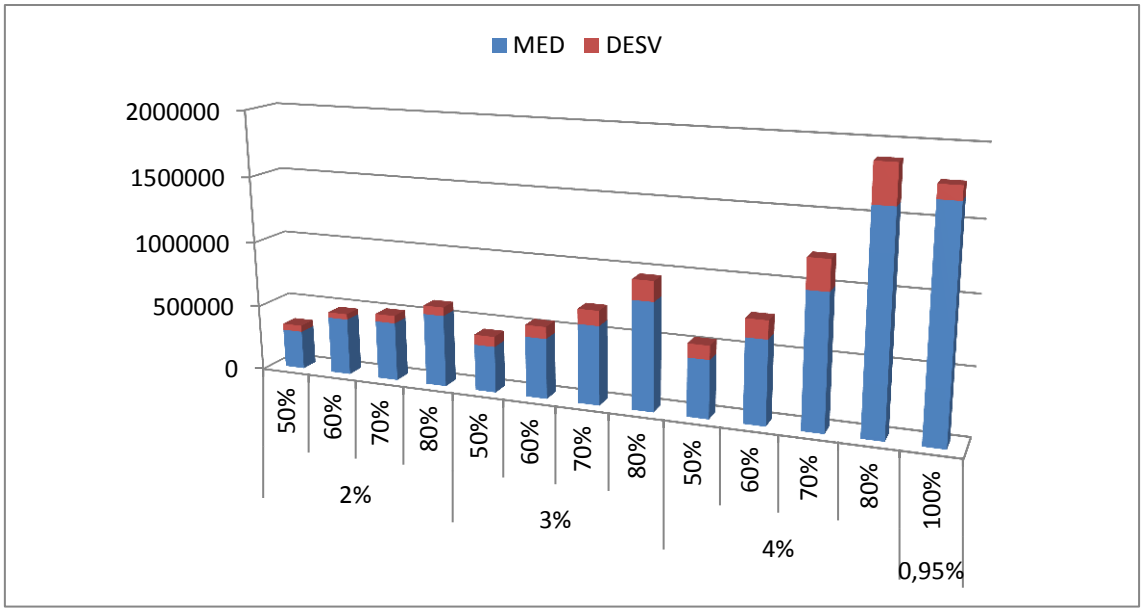

Figura 4.24: Comparación de resultados (nº iteraciones) QAP 80

4.3.3. Análisis de los resultados

A continuación, se va a evaluar el comportamiento del algoritmo (*la influencia de la temperatura inicial y el parámetro* α *en la eficiencia del algoritmo*) en los diferentes escenarios planteados – cuando el algoritmo se ejecuta con una temperatura inicial correspondiente a porcentajes entre el 50% y el 80% del número de ciclos de enfriamiento de la temperatura inicial estándar, y la variación dinámica del parámetro α se realiza con incrementos porcentuales del 2%, 3%, y 4% –.

Prueba estadística

En el estudio se van a realizar sendas estimaciones estadísticas por intervalo al 95% de confianza, para la diferencia de objetivos medios y la diferencia de iteraciones medias respectivamente, tomando como referencia el objetivo medio y el número medio de iteraciones obtenidos con el enfriamiento estándar. Con ello se debe verificar:

- Cuál es la combinación de parámetros para los que el objetivo medio no es significativamente diferente (mejor/peor) que el obtenido en el caso estándar.

- Cuál es la combinación de parámetros para los que el número medio de iteraciones no es significativamente distinto (menor/mayor) que el obtenido en el caso estándar.

Para analizar si puede inferirse una diferencia significativa se va a realizar una estimación por intervalo de confianza para la diferencia de medias, tanto de objetivo como del número de iteraciones. En cada caso se dispone de dos muestras aleatorias independientes de tamaño n_1 y n_2 respectivamente. El objetivo es comparar dos medias poblacionales μ_1 y μ_2. Para ello, lo adecuado es basarse en el estimador $\bar{x}_1 - \bar{x}_2$, diferencia entre ambas medias muestrales.

La situación que se estudia, en esta comparación, es la de dos muestras aleatorias independientes del mismo tamaño y suficientemente grandes, de manera que puede asegurarse la normalidad aproximada de ambas medias muestrales.

Se analizará si existe una diferencia significativa entre el objetivo medio conseguido con el enfriamiento para cada porcentaje de variación dinámica del parámetro α y de número de ciclos de temperatura y el objetivo medio obtenido con el enfriamiento estándar. Igualmente se procederá en el estudio del número medio de iteraciones. Como las varianzas poblacionales son finitas y desconocidas, las sustituiremos por las cuasivarianzas muestrales S_1^2 y S_2^2, dando lugar al intervalo:

$$\left(-z_{\alpha/2} \leq Z \leq z_{\alpha/2} \right)$$

$$Z = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$$

- Si $Z < -z_{\alpha/2}$, es mejor el caso estándar
- Si $Z > z_{\alpha/2}$, el caso estándar ofrece soluciones de peor calidad
- Si $\left(-z_{\alpha/2} \leq Z \leq z_{\alpha/2}\right)$, la prueba no permite afirmar que existan diferencias significativas entre ambos casos

$\bar{x}_1$ = *media muestral (caso estándar)*

$\bar{x}_2$ = *media muestral (caso estándar)*^{c}

n_1 = *tamaño muestra (caso estándar)*

n_2 = *tamaño muestra (caso estándar)*^{c}

S_1 = *cuasivarianza muestral (caso estándar)*

S_2 = *cuasivarianza muestral (caso estándar)*^{c}

Comparación entre el enfriamiento con variación dinámica del parámetro α del 2% en combinación con el 50% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=2% T_0=50%	Objetivo	7160,86	329,58

$$Z = -2{,}39 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=2% T_0=50%	Objetivo	35322,18	1548,06

$$(-1{,}96 \leq 0{,}05 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=2% T_0=50%	Objetivo	26533714	54019,71

$$Z = -3{,}41 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=2% T_0=50%	Objetivo	250771702	333069,44

$$Z = -4{,}17 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.39: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar

- En TSP 47 y en las dos instancias del problema QAP se puede considerar que el enfriamiento estándar proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 2% en combinación con un porcentaje del número de ciclos de temperatura del 50%.

- En TSP 80 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=2% T_0=50%	Iteraciones	85263,6	14869,27

$$Z = 127{,}78 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=2% T_0=50%	Iteraciones	317300,31	60728,38

$$Z = 129{,}91 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=2% T_0=50%	Iteraciones	100409,47	19488,76

$$Z = 106{,}48 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=2% T_0=50%	Iteraciones	293557,53	48569,93

$$Z = 122{,}61 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.40: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 2% en combinación con un porcentaje del número de ciclos de temperatura del 50% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 2% en combinación con el 60% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=2% T_0=60%	Objetivo	7182,15	386,27

$$Z = -2{,}61 < -1{,}96 = -z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=2% T_0=60%	Objetivo	35609,04	1432,79

$$(-1{,}96 \leq -1{,}29 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=2% T_0=60%	Objetivo	26536665	52746,71

$$Z = -3{,}9 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=2% T_0=60%	Objetivo	250700410	342584,86

$$Z = -2{,}63 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.41: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar

- En TSP 47 y en las dos instancias del problema QAP se puede considerar que el enfriamiento estándar proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 2% en combinación con un porcentaje del número de ciclos de temperatura del 60%.

- En TSP 80 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=2% T_0=60%	Iteraciones	103296,05	14083,15

$$Z = 123{,}56 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=2% T_0=60%	Iteraciones	407841,97	82278,72

$$Z = 108{,}94 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=2% T_0=60%	Iteraciones	123206,16	22438,56

$$Z = 98{,}09 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=2% T_0=60%	Iteraciones	432107,22	42287,57

$$Z = 112{,}82 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.42: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 2% en combinación con un porcentaje del número de ciclos de temperatura del 60% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 2% en combinación con el 70% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=2% T_0=70%	Objetivo	7101,99	352,66

$$(-1{,}96 \leq -1{,}18 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=2% T_0=70%	Objetivo	35381,77	1507,14

$$(-1{,}96 \leq -0{,}21 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=2% T_0=70%	Objetivo	26527046	46819,24

$$Z = -2{,}64 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=2% T_0=70%	Objetivo	250690324	356043,96

$$Z = -2{,}37 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.43: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar

- En las dos instancias del problema TSP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.
- En las dos instancias del problema QAP se puede considerar que el enfriamiento estándar proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 2% en combinación con un porcentaje del número de ciclos de temperatura del 70%.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=2% T_0=70%	Iteraciones	123405,22	13866,41

$$Z = 117{,}62 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=2% T_0=70%	Iteraciones	492320,94	75096,85

$$Z = 106{,}09 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=2% T_0=70%	Iteraciones	149331,18	26519,91

$$Z = 88{,}01 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=2% T_0=70%	Iteraciones	444331,99	58282,2

$$Z = 104{,}84 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.44: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 2% en combinación con un porcentaje del número de ciclos de temperatura del 70% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 2% en combinación con el 80% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=2% T_0=80%	Objetivo	7116,13	347,319

$$(-1{,}96 \leq -1{,}47 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=2% T_0=80%	Objetivo	35254,95	1567,55

$$(-1{,}96 \leq 0{,}35 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=2% T_0=80%	Objetivo	26530536	46846,33

$$Z = -3{,}2 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=2% T_0=80%	Objetivo	250676638	372644,25

$$Z = -2{,}03 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.45: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar

- En las dos instancias del problema QAP se puede considerar que el enfriamiento estándar proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 2% en combinación con un porcentaje del número de ciclos de temperatura del 80%.

- En las dos instancias del problema TSP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=2% T_0=80%	Iteraciones	144518,15	15252,51

$$Z = 108{,}86 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=2% T_0=80%	Iteraciones	601318,09	92365,27

$$Z = 88{,}46 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=2% T_0=80%	Iteraciones	182145,71	25363,06

$$Z = 82{,}16 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=2% T_0=80%	Iteraciones	542629,24	65429,78

$$Z = 93{,}38 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.46: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 2% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 2% en combinación con un porcentaje del número de ciclos de temperatura del 80% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 3% en combinación con el 50% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=3% T_0=50%	Objetivo	7082,8	363,31

$$(-1{,}96 \leq -0{,}8 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=3% T_0=50%	Objetivo	35466,55	1516

$$(-1{,}96 \leq -0{,}6 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=3% T_0=50%	Objetivo	26537054	48008,02

$$Z = -4{,}18 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=3% T_0=50%	Objetivo	250750833	390452,76

$$Z = -3{,}04 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.47: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar

- En las dos instancias del problema TSP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.
- En las dos instancias del problema QAP se puede considerar que el enfriamiento estándar proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 50%.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=3% T_0=50%	Iteraciones	94147,07	17670,41

$$Z = 119{,}92 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=3% T_0=50%	Iteraciones	386736,41	91062,19

$$Z = 105{,}56 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=3% T_0=50%	Iteraciones	123200,96	28136,52

$$Z = 91{,}75 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=3% T_0=50%	Iteraciones	349686,11	76558,72

$$Z = 103{,}91 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.48: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 50% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 3% en combinación con el 60% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=3% T_0=60%	Objetivo	7076,37	338,31

$$(-1{,}96 \leq -0{,}71 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=3% T_0=60%	Objetivo	35250,82	1492,81

$$(-1{,}96 \leq 0{,}38 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=3% T_0=60%	Objetivo	26520634	48570,62

$$(-1{,}96 \leq -1{,}58 \leq 1{,}96)$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=3% T_0=60%	Objetivo	250604631	351909,83

$$(-1{,}96 \leq -0{,}62 \leq 1{,}96)$$

Tabla 4.49: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=3% T_0=60%	Iteraciones	120736,86	20801,18

$$Z = 106{,}53 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=3% T_0=60%	Iteraciones	504103,8	101378,57

$$Z = 91{,}35 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=3% T_0=60%	Iteraciones	154990,16	35562,9

$$Z = 77{,}54 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=3% T_0=60%	Iteraciones	454004,4	89624,66

$$Z = 89{,}75 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.50: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 60% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 3% en combinación con el 70% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=3% T_0=70%	Objetivo	7045,66	293,32

$$(-1{,}96 \leq -0{,}11 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=3% T_0=70%	Objetivo	35569,71	1539,51

$$(-1{,}96 \leq -1{,}06 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=3% T_0=70%	Objetivo	26527358	49654,78

$$Z = -2{,}61 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=3% T_0=70%	Objetivo	250661039	379075,09

$$(-1{,}96 \leq -1{,}71 \leq 1{,}96)$$

Tabla 4.51: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=3% T_0=70%	Iteraciones	155209,37	23034,48

$$Z = 93{,}25 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=3% T_0=70%	Iteraciones	686269,19	117545,95

$$Z = 71{,}71 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=3% T_0=70%	Iteraciones	201693,74	46989,12

$$Z = 59{,}34 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=3% T_0=70%	Iteraciones	592162,09	112452,62

$$Z = 70{,}99 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.52: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 70% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 3% en combinación con el 80% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=3% T_0=80%	Objetivo	7087,48	352,33

$$(-1{,}96 \leq -0{,}91 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=3% T_0=80%	Objetivo	35883,19	1670,77

$$Z = -2{,}39 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=3% T_0=80%	Objetivo	26526850	47194,62

$$Z = -2{,}61 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=3% T_0=80%	Objetivo	250639704	387407,99

$$(-1{,}96 \leq -1{,}27 \leq 1{,}96)$$

Tabla 4.53: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar

- En TSP 47 y QAP 80 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.
- En TSP 80 y QAP 47 se puede considerar que el enfriamiento estándar proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 80%.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=3% T_0=80%	Iteraciones	202519,2	31268,65

$$Z = 69{,}72 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=3% T_0=80%	Iteraciones	984062,77	236338,04

$$Z = 30{,}61 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=3% T_0=80%	Iteraciones	267612,43	60211,27

$$Z = 41{,}3 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=3% T_0=80%	Iteraciones	808311,64	150395,21

$$Z = 47{,}25 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.54: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 3% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 80% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 4% en combinación con el 50% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=4% T_0=50%	Objetivo	7113,98	359,79

$$(-1{,}96 \leq -1{,}4 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=4% T_0=50%	Objetivo	35900,67	1668,19

$$Z = -2{,}46 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=4% T_0=50%	Objetivo	26535772	51233,42

$$Z = -3{,}83 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=4% T_0=50%	Objetivo	250673195	358630,69

$$Z = -2{,}01 < -1{,}96 = -z_{\alpha/2}$$

Tabla 4.55: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar

- En TSP 47 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.
- En TSP 80 y en las dos instancias del problema QAP se puede considerar que el enfriamiento estándar proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 4% en combinación con un porcentaje del número de ciclos de temperatura del 50%.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=4% T_0=50%	Iteraciones	112386,92	21081,71

$$Z = 108{,}38 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=4% T_0=50%	Iteraciones	533943,92	183432,23

$$Z = 59{,}67 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=4% T_0=50%	Iteraciones	143296,17	41657,77

$$Z = 73{,}75 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=4% T_0=50%	Iteraciones	433985,18	103053,04

$$Z = 85{,}36 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.56: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 50%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 4% en combinación con un porcentaje del número de ciclos de temperatura del 50% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 4% en combinación con el 60% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=4% T_0=60%	Objetivo	7129,09	354,93

$$(-1{,}96 \leq -1{,}7 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=4% T_0=60%	Objetivo	35507,61	1646,57

$$(-1{,}96 \leq -0{,}75 \leq 1{,}96)$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=4% T_0=60%	Objetivo	26521725	51988,64

$$(-1{,}96 \leq -1{,}69 \leq 1{,}96)$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=4% T_0=60%	Objetivo	250655350	377946,99

$$(-1{,}96 \leq -1{,}6 \leq 1{,}96)$$

Tabla 4.57: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=4% T_0=60%	Iteraciones	156160,98	28824,83

$$Z = 84{,}14 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=4% T_0=60%	Iteraciones	800911,16	232461,12

$$Z = 38{,}36 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=4% T_0=60%	Iteraciones	205491,96	55262,08

$$Z = 53 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=4% T_0=60%	Iteraciones	624125,01	135797,41

$$Z = 61{,}49 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.58: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 60%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 4% en combinación con un porcentaje del número de ciclos de temperatura del 60% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 4% en combinación con el 70% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=4% T_0=70%	Objetivo	7109,48	396,59

$$(-1{,}96 \leq -1{,}26 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=4% T_0=70%	Objetivo	35805,83	1527,68

$$Z = -2{,}14 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=4% T_0=70%	Objetivo	26524852	48578,17

$$Z = -2{,}24 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=4% T_0=70%	Objetivo	250596614	330386,1

$$(-1{,}96 \leq -0{,}47 \leq 1{,}96)$$

Tabla 4.59: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar

- En TSP 47 y en QAP 80 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.
- En TSP 80 y en QAP 47 se puede considerar que el enfriamiento estándar proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 4% en combinación con un porcentaje del número de ciclos de temperatura del 70%.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=4% T_0=70%	Iteraciones	226892,92	50520,61

$$Z = 46{,}74 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=4% T_0=70%	Iteraciones	1343796,5	339716,66

$$Z = 11{,}86 > 1{,}96 = z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=4% T_0=70%	Iteraciones	303450,15	71597,87

$$Z = 31{,}93 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=4% T_0=70%	Iteraciones	995925,57	223978,59

$$Z = 27{,}18 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.60: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 70%) y el enfriamiento estándar

- En las dos instancias de los problemas TSP y QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 4% en combinación con un porcentaje del número de ciclos de temperatura del 70% converge más rápido que el algoritmo con enfriamiento estándar.

Comparación entre el enfriamiento con variación dinámica del parámetro α del 4% en combinación con el 80% de ciclos de temperatura y el enfriamiento estándar.

TSP 47:

		MED	DESV
100%	Objetivo	7040,17	383,07

		MED	DESV
A=4% T_0=80%	Objetivo	7068,74	384,24

$$(-1{,}96 \leq -0{,}52 \leq 1{,}96)$$

TSP 80:

		MED	DESV
100%	Objetivo	35334,72	1574,21

		MED	DESV
A=4% T_0=80%	Objetivo	35996,34	1803,34

$$Z = -2{,}76 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Objetivo	26510487,2	41489,7

		MED	DESV
A=4% T_0=80%	Objetivo	26524470	52923,73

$$Z = -2{,}07 < -1{,}96 = -z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Objetivo	250574322,1	335723,84

		MED	DESV
A=4% T_0=80%	Objetivo	250519437	274385,22

$$(-1{,}96 \leq 1{,}26 \leq 1{,}96)$$

Tabla 4.61: Intervalo de confianza para la diferencia de objetivos medios entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar

- En TSP 80 QAP 47 se puede considerar que el enfriamiento estándar proporciona mejores soluciones que el enfriamiento con variación dinámica del parámetro α del 4% en combinación con un porcentaje del número de ciclos de temperatura del 80%.
- En TSP 47 y QAP 80 la prueba no permite afirmar que existan diferencias significativas en la calidad de las soluciones entre ambos casos.

TSP 47:

		MED	DESV
100%	Iteraciones	498561,6	28723,43

		MED	DESV
A=4% T_0=80%	Iteraciones	343455,28	62687,19

$$Z = 22{,}49 > 1{,}96 = z_{\alpha/2}$$

TSP 80:

		MED	DESV
100%	Iteraciones	1761604,4	93123,83

		MED	DESV
A=4% T_0=80%	Iteraciones	2142448,5	297859,62

$$Z = -12{,}2 < -1{,}96 = -z_{\alpha/2}$$

QAP 47:

		MED	DESV
100%	Iteraciones	563650,78	38893,8

		MED	DESV
A=4% T_0=80%	Iteraciones	518908,9	109670,15

$$Z = 3{,}84 > 1{,}96 = z_{\alpha/2}$$

QAP 80:

		MED	DESV
100%	Iteraciones	1663308,76	100600,05

		MED	DESV
A=4% T_0=80%	Iteraciones	1599538,5	285546,93

$$Z = 2{,}1 > 1{,}96 = z_{\alpha/2}$$

Tabla 4.62: Intervalo de confianza para la diferencia de medias de iteraciones entre el enfriamiento adaptativo (variación dinámica del parámetro α del 4% y porcentaje del número de ciclos de temperatura del 80%) y el enfriamiento estándar

- En TSP 47 y las dos instancias del problema QAP se puede considerar que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 4% en combinación con un porcentaje del número de ciclos de temperatura del 80% converge más rápido que el algoritmo con enfriamiento estándar.

- En TSP 80 se puede considerar que el algoritmo con enfriamiento estándar converge más rápido que el de enfriamiento con variación dinámica del parámetro α del 4% en combinación con un porcentaje del número de ciclos de temperatura del 80%.

A continuación, se recoge en forma resumida la información de la comparativa en sendas tablas, una sobre objetivos y otra sobre convergencia temporal.

En la tabla 4.63, el símbolo '+' significa que el algoritmo con enfriamiento estándar da mejores soluciones, '=' quiere decir que la diferencia no es significativa, y '-' significa que el algoritmo con enfriamiento estándar ofrece soluciones de peor calidad.

	2%				3%			
Instancia	**50%**	**60%**	**70%**	**80%**	**50%**	**60%**	**70%**	**80%**
TSP 47	+	+	=	=	=	=	=	=
TSP 80	=	=	=	=	=	=	=	+
QAP 47	+	+	+	+	+	=	+	+
QAP 80	+	+	+	+	+	=	=	=

Tabla 4.63: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de objetivos según la distribución normal.

	4%			
Instancia	**50%**	**60%**	**70%**	**80%**
TSP 47	=	=	=	=
TSP 80	+	=	+	+
QAP 47	+	=	+	+
QAP 80	+	=	=	=

Tabla 4.63 (cont): Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de objetivos según la distribución normal.

En la tabla 4.64, '+' significa que el algoritmo con enfriamiento estándar converge más rápido, '=' quiere decir que la diferencia no es significativa, y '-' significa que el algoritmo con enfriamiento estándar converge más lentamente.

	2%				3%			
Instancia	**50%**	**60%**	**70%**	**80%**	**50%**	**60%**	**70%**	**80%**
TSP 47	-	-	-	-	-	-	-	-
TSP 80	-	-	-	-	-	-	-	-
QAP 47	-	-	-	-	-	-	-	-
QAP 80	-	-	-	-	-	-	-	-

Tabla 4.64: Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de número de iteraciones según la distribución normal

	4%			
Instancia	**50%**	**60%**	**70%**	**80%**
TSP 47	-	-	-	-
TSP 80	-	-	-	+
QAP 47	-	-	-	-
QAP 80	-	-	-	-

Tabla 4.64 (cont): Resumen de la comparativa mediante intervalo de confianza al 95% para la diferencia de medias de número de iteraciones según la distribución normal

Conclusiones

Con respecto al estudio combinado de la influencia de la temperatura inicial y el parámetro α en la eficiencia del algoritmo, analizando los resultados y gráficas obtenidos, puede observarse que cuanto mayor es el porcentaje de incremento de α y mayor es el porcentaje del número de ciclos de enfriamiento respecto al total de ciclos para la temperatura inicial estándar, mayor es el tiempo de ejecución del algoritmo, ya que la temperatura desciende más lentamente, y como era previsible, cuanto menor es el porcentaje del número de ciclos de enfriamiento respecto al total de ciclos para la temperatura inicial estándar peores son las soluciones obtenidas.

Tras el análisis estadístico realizado, se puede concluir con respecto al tiempo de ejecución que, para las dos instancias de los problemas TSP y QAP, el algoritmo con enfriamiento adaptativo en todas sus combinaciones converge más rápidamente que el algoritmo con enfriamiento estándar, exceptuando el caso del TSP 80 con porcentaje de incremento de α del 4% y porcentaje del número de ciclos de temperatura del 80%.

Con respecto a la calidad del objetivo, la combinación de parámetros para los que el objetivo medio no es significativamente diferente (peor) que el obtenido en el caso estándar, se produce en dos casos:

- Porcentaje de incremento de α del 3% y porcentaje del número de ciclos de temperatura del 60%.
- Porcentaje de incremento de α del 4% y porcentaje del número de ciclos de temperatura del 60%.

Si bien, en el primer caso la convergencia es más rápida.

En cuanto a la desviación típica del objetivo, la obtenida con el enfriamiento adaptativo combinado no es significativamente diferente (peor) que la obtenida con el enfriamiento estándar en ninguno de los problemas tratados. En las dos instancias del problema TSP la desviación del objetivo mejora con respecto al caso estándar y en las dos instancias del problema QAP es similar. Por tanto, puede afirmarse que la variante adaptativa con porcentaje del 2% de

incremento del parámetro α combinado con porcentaje del 60% del número de ciclos de temperatura constituye un algoritmo más robusto que el del enfriamiento estándar.

La principal conclusión que puede extraerse de la comparativa realizada es que el algoritmo con enfriamiento adaptativo basado en la variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 60% proporciona un objetivo medio y una desviación típica del objetivo estadísticamente similares a los del enfriamiento estándar, con una mayor reducción del tiempo de ejecución (iteraciones).

4.4. ANÁLISIS CONJUNTO DE LAS PRUEBAS REALIZADAS

El recocido simulado, como se ha mencionado previamente, es una de las más importantes meta-heurísticas, siendo bien conocidas sus propiedades de convergencia hacia soluciones de alta calidad aunque con un elevado tiempo computacional.

Tras el análisis estadístico realizado en los tres estudios, las conclusiones extraídas son:

1. *En la influencia de la temperatura inicial en el comportamiento del algoritmo*, se puede concluir que para un porcentaje de ciclos de enfriamiento igual o superior al 50% la calidad de la solución media obtenida (objetivo) no es significativamente diferente (peor) con respecto a la obtenida con la temperatura inicial estándar.

2. *En la influencia del parámetro* α *cuando éste se modifica dinámicamente en la ecuación de reducción exponencial de la temperatura*, se puede concluir que el comportamiento del algoritmo con un porcentaje de variación dinámica del 2%, proporciona un objetivo estadísticamente similar con una mayor reducción del tiempo de ejecución (iteraciones) respecto al caso estándar.

3. *En la influencia combinada de la temperatura inicial y el parámetro* α *en la eficiencia del algoritmo,* se puede concluir que el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 60% mantiene un objetivo medio y una desviación típica del objetivo estadísticamente similares a los del algoritmo estándar reduciendo el tiempo de ejecución (iteraciones).

El objetivo de los tres estudios es determinar la combinación de parámetros que, manteniendo un objetivo medio y una desviación típica del objetivo estadísticamente similares a los del algoritmo estándar, consigan reducir al máximo el tiempo de ejecución (iteraciones).

A la vista de los resultados obtenidos, se concluye que la mejor solución corresponde al estudio tercero, consistente en la evaluación combinada de la influencia de la temperatura inicial y el parámetro α en la eficiencia del algoritmo. Es decir, el algoritmo basado en enfriamiento con variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 60% mantiene un objetivo medio y una desviación típica del objetivo estadísticamente similares a los del algoritmo estándar reduciendo al máximo el tiempo de ejecución (iteraciones). De hecho, la reducción del tiempo de ejecución es muy notable: el enfriamiento adaptativo óptimo requiere el 30% del tiempo medio que emplea el enfriamiento estándar con los valores usuales de los parámetros.

Capítulo 5: Una aplicación práctica de la Estrategia de Enfriamiento Adaptativo: Problema de Selección de Carteras de Inversión

"Innovar es encontrar nuevos o mejorados usos a los recursos de que ya disponemos"

Peter Drucker

5.1. EL PROBLEMA DE LA SELECCIÓN DE CARTERAS DE INVERSIÓN

En este capítulo, se presentan los fundamentos en el mundo financiero del Modelo de Markowitz [MARK52] y, que hoy en día, suponen la base teórica moderna de la selección de carteras de inversión. Markowitz inició una revolución al sugerir que la mejor manera de evaluar un activo es mediante el cálculo de la media y varianza de sus rendimientos, así como su correlación con respecto a otros activos de la cartera. Explica que el inversor, en su empeño por elegir aquellos activos que sean más prometedores, va a invertir en distintos activos financieros cuyos comportamientos son inciertos. Por esta razón, propone abordar la cartera como un único elemento del cual deben estudiarse las características de riesgo y de rendimiento global en lugar de tratarse por separado.

Las carteras de inversión ofrecen la posibilidad de obtener mayores rendimientos que los logrados invirtiendo en instrumentos financieros de forma individual. Las carteras son un conjunto de valores financieros, en donde cada valor tiene asociado un riesgo y un beneficio. La teoría de selección de instrumentos financieros para la creación de carteras de inversión desarrolla la idea de la diversificación de los instrumentos financieros [MARK70] partiendo

del hecho de que si se combinan diferentes riesgos y beneficios se obtiene en suma un equilibrio de las pérdidas y ganancias, lo que sería complicado de obtener si solamente se invirtiera en un sólo valor. La combinación de los diferentes activos de inversión da como resultado una gran variedad de posibilidades, obteniéndose diferentes carteras. El problema de Optimización de Carteras de Inversión se ha seleccionado como aplicación práctica en esta tesis y, establece la proporción que debe ser invertida en cada tipo de activo, de manera que se maximice el rendimiento para un cierto riesgo dado o bien que minimice el riesgo para un rendimiento esperado.

En el ámbito financiero surgen problemas para cuya resolución es necesario realizar un proceso de optimización. Problemas que requieren determinadas restricciones que nacen a partir de los propios mercados financieros, los inversores, las transacciones, factores externos que pueden afectar al mercado en general como es el entorno económico nacional e internacional, etc.. Además, cada inversor tendrá su propia valoración de lo que desea obtener, del riesgo que está dispuesto a asumir, el plazo de la inversión, los dividendos, etc. La mayoría de estos problemas de optimización son difíciles de resolver en la práctica, entendiéndose por problema de optimización difícil de resolver aquel para el que no se puede encontrar la solución óptima en un tiempo razonable.

La complejidad para resolver este problema de optimización, Problema de la Selección de Carteras de Inversión (*Portfolio Investment Problem, PIP*), se encuentra en el tipo de limitaciones que contiene y de los activos a incluir en la cartera. Este problema ha servido como referencia en sus versiones más simplificadas o académicas para el estudio de nuevas técnicas de optimización en las últimas décadas, por lo que se considera un problema fundamental de la optimización combinatoria.

Diferentes investigadores han buscado fabricar modelos más objetivos que reflejen la complejidad de los mercados financieros, como es el problema planteado por los investigadores Crama y Schyns [CRAM01] que utilizaron heurísticas para encontrar soluciones al Modelo extendido de Markowitz.

5.1.1. Teoría Moderna de Carteras

El Problema de la Selección de Carteras de Inversión consiste en construir una cartera formada por un conjunto de activos financieros en que se divide la inversión de un individuo o una institución.

La construcción de una cartera de inversión [REIL97] equivale a realizar la selección adecuada de los activos de inversión que la integrarán, así como también determinar la proporción de dicha inversión que se destinará a cada uno de estos activos.

Como ejemplo, en la figura 5.1, se muestra una cartera de inversión formada por seis activos y el porcentaje de la cartera invertida en cada activo.

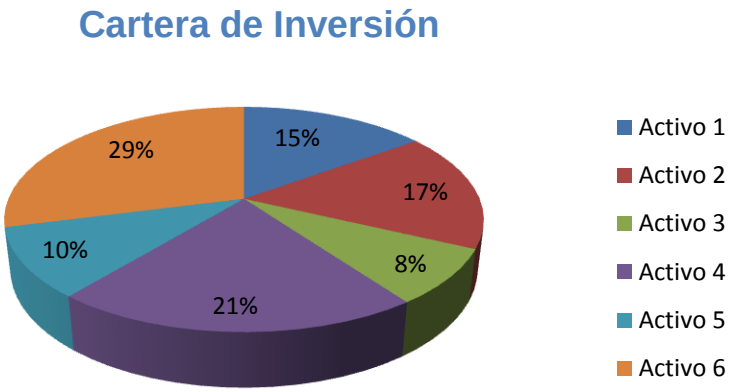

Figura 5.1: Ejemplo de una cartera de inversión

Cuando se construye una cartera hay que analizar dos conceptos básicos – *la rentabilidad y el riesgo* –. Expresado de otra manera, beneficios que se obtienen en función de la inversión y el riesgo que supone la inversión con respecto al resultado de la inversión. Una *cartera de inversión de valores* se define como la distribución de un capital en diferentes opciones de inversión. Una combinación de activos o títulos individuales ofrecen un riesgo menor con respecto a un activo o título individual. La determinación de la estrategia óptima de inversión para maximizar los beneficios de su capital final en un periodo, se conoce como *Optimización de una cartera de valores de inversión*.

El proceso de seleccionar activos financieros entre la gran variedad existente y establecer la proporción de la inversión a cada uno los seleccionados con el objetivo de optimizar los beneficios, no es un proceso simple.

Los fundamentos básicos más utilizados para tomar decisiones en el diseño de las carteras se encuentran en el Modelo de Markowitz [MARK52] y, que hoy en día, suponen la base teórica moderna de la selección de carteras de inversión. A partir de la publicación de estos fundamentos, el problema de selección de carteras [MARK59] se ha convertido en uno de las líneas más extendidas de investigación en el mundo financiero.

Harry Markowitz, premio Nobel de Economía en el año 1990, planteó que la manera de caracterizar un activo financiero es mediante el cálculo de la media y la varianza de sus rendimientos, así como su comportamiento con respecto a otros activos incluidos en la cartera. Bajo esta premisa, plantea estudiar la cartera como un único elemento del cual deben analizarse el riesgo y el rendimiento global en lugar de tratarse por separado.

El proceder racional de un inversor se caracteriza por componer una cartera que genere la máxima rentabilidad para un cierto riesgo, o bien, minimizar el riesgo para una rentabilidad dada**.** Las investigaciones de Markowitz se centraron en la construcción óptima de carteras y cómo se puede reducir el riesgo de las rentabilidades optando por activos de diferente comportamiento. La idea central del modelo es reducir el riesgo en la inversión a través de dividir el capital en diferentes activos. Un inversor puede obtener un mayor beneficio de su cartera por medio de la diversificación. Gracias a este concepto, se puede seleccionar un número de activos que por sí solos serían catalogados como de mayor riesgo, pero que, formando parte de una combinación de activos, obtienen un mayor potencial de rentabilidad. Si bien existen beneficios en la diversificación, el riesgo de una cartera no puede eliminarse totalmente, sólo se puede minimizar.

En el planteamiento del Modelo de Markowitz, se trabaja con carteras basadas en un sólo periodo, donde únicamente en el inicio se decide sobre la distribución del capital entre los diferentes activos.

Se establecen otras premisas tales como:

- La inversión es integral.
- Los inversores tienen la misma información.
- Los activos son infinitamente divisibles.
- Las decisiones de compra y venta de valores no afectan el mercado.
- No existen impuestos ni costes de transacción.
- Las decisiones se basan exclusivamente en los rendimientos esperados y el riesgo.
- Aversión al riesgo, significa que para un nivel de rendimiento esperado, los inversores prefieren el mínimo riesgo posible y ante un nivel de riesgo dado, se decantarán por un mayor rendimiento.

La teoría de Markowitz se sustenta en dos hipótesis**:**

- La rentabilidad de una cartera, es una variable aleatoria cuya distribución de probabilidad para un sólo periodo es conocida (únicamente en el inicio se decide sobre la distribución del capital en los diferentes valores). El beneficio en la inversión se caracteriza por la *esperanza matemática* de la variable aleatoria.
- El riesgo en la inversión se caracteriza por la varianza, o por la desviación típica de la variable aleatoria que describe la rentabilidad de una cartera. La mínima varianza se obtiene con carteras bien diversificadas.

La relación existente entre rentabilidad y riesgo, y cómo esta correlación afecta la composición de una cartera, es el principal concepto que todo inversor debe conocer. Los esfuerzos en la construcción de una cartera de valores se deben centrar en distribuir óptimamente la inversión entre diferentes activos con la idea esencial de diversificación. La diversificación es la mejor herramienta

contra el riesgo. Esto se consigue combinando activos que maximicen la rentabilidad para un riesgo esperado, o bien, minimizan el riesgo para una rentabilidad determinada. A este modelo, planteado por Markowitz, se le conoce como *Teoría Media-Varianza* o *Teoría Moderna de Carteras (Modern Portfolio Theory).*

5.1.2. Rendimiento y Riesgo de una Cartera

El rendimiento esperado de una cartera *p* es una media ponderada de las rentabilidades esperadas para los distintos activos individuales que comprenden la cartera.

A continuación, se van a definir la rentabilidad de un activo en un periodo de tiempo; la rentabilidad media de un activo en un periodo de un tiempo; y la rentabilidad de una cartera formada por un conjunto de activos.

Rentabilidad de un activo en un periodo

La rentabilidad individual b_i (en tanto por uno) de un activo *i* en un período de tiempo *t* se define por:

$$b_i = \frac{P_{it} - P_{it-1}}{P_{it-1}}$$

P_{it} es el precio del activo *i* al final del período *t* , y

P_{it-1} es el precio del activo *i* al final del periodo *t–1*, o lo que es lo mismo, el precio del activo *i* al comienzo del período *t*.

Rentabilidad media de un activo en un periodo

La rentabilidad media $\overline{b_i}$ de un activo *i* en un período de tiempo *t* se define por:

$$\overline{b_i} = \frac{1}{T}\sum_{t=1}^{T} b_{it}$$

T indica el número de períodos de tiempo con el histórico de inversiones,

La varianza media de un activo *i* en un período de tiempo *t* se define por:

$$\sigma_i{}^2 = \frac{1}{T-1}\sum_{t=1}^{T}\left(b_{it} - \overline{b_i}\right)^2$$

Como una cartera está compuesta por diferentes activos, la varianza de la cartera no depende sólo de cada una de las varianzas individuales, sino también de la covarianza de los rendimientos entre cada par de activos, es decir, el riesgo de una cartera no es la suma de los riesgos de los valores que la componen, sino que existe otra variable relacionada al riesgo total y esta es la covarianza de los rendimientos. La covarianza indica cuál será el comportamiento de un activo al producirse una variación en el valor de otro activo. Se define como:

$$\sigma_{ij} = \frac{\sum_{t=1}^{T}\left(b_{it} - \overline{b_i}\right)\left(b_{jt} - \overline{b_j}\right)}{T-1}$$

El valor de la covarianza puede ser:

- Positivo; *cuando una acción sube, la otra también tiende a subir*
- Negativo; *cuando uno sube, el otro tiende a bajar*
- Próximo a cero; *probablemente ambos activos no estén relacionados*

De la misma forma que existe un vector de rendimientos esperados $\boldsymbol{b} = (b_1,\ldots,b_n)$, respecto al riesgo de una cartera, existe una matriz de covarianzas:

$$\left[\sigma_{ij}\right] = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1n} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_n^2 \end{pmatrix}$$

Rentabilidad de una cartera formada por n activos

La rentabilidad de una cartera o rendimiento esperado formada por *n* activos $E\left[R_p\right]$, se obtiene como la suma de rentabilidades por la proporción de cada activo *i* contenido en la cartera:

$$E\left[R_p\right] = b_1 x_1 + \ \ldots\ + b_n x_n \ = \sum_{i=1}^{n} \overline{b_i} x_i$$

donde b_i y x_i representan el rendimiento y el porcentaje (en tanto por uno) de la cartera invertida en el activo *i.*

El riesgo se representa a través de la varianza o con la desviación típica y representa el grado de dispersión en torno a la media del rendimiento del activo. Una distribución con rendimientos sumamente volátiles de un período indica una elevada incertidumbre (riesgo) sobre el rendimiento posible de la inversión realizada. Por consiguiente, cuanto mayor sea la varianza de la distribución de probabilidades de los posibles rendimientos de una inversión, menos interesante resultará la cartera.

La varianza de una cartera se calcula como:

$$\sigma^2(R_p) = x_1^2\sigma_1^2 + \ldots + x_n^2\sigma_n^2 + 2x_1x_2\sigma_{12} + \ldots + 2x_{n\text{-}1}x_n\sigma_{(n-1)n} = \sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{ij}x_i x_j$$

donde *n* es el número de activos que constituyen la cartera, x_i es el porcentaje (en tanto por uno) de la cartera invertida en el *i*, σ_i la desviación típica del rendimiento del activo *i* y σ_{ij} es la covarianza del rendimiento del activo *i* con el rendimiento del activo *j*.

5.1.3. Frontera Eficiente

En función de la variación de la proporción x_i de la cartera invertida en el activo *i*, se pueden obtener infinitas carteras. Si el conjunto de pares $(E\lfloor R_p \rfloor, \sigma^2(R_p))$ o combinaciones rentabilidad/riesgo se colocan en una gráfica (ver figura 5.1) donde el eje de abscisas es el riesgo de la cartera y el eje de ordenadas su rendimiento, se obtiene una nube de puntos en las que se podrán observar carteras con mejor relación rendimiento/riesgo que otras.

El conjunto de estas carteras eficientes – *aquellas que dan menor riesgo para una determinada rentabilidad media* – se denomina Frontera Eficiente. Una vez conocida ésta, el inversor elegirá su cartera óptima, en función de sus preferencias.

Se define Frontera Eficiente como el subconjunto de puntos que representan la mínima varianza y por lo tanto caracterizan el conjunto de carteras con menor riesgo para una determinada rentabilidad media.

El problema que se plantea es la de decidir entre los diferentes puntos del mapa de oportunidades posibles [GOME91].

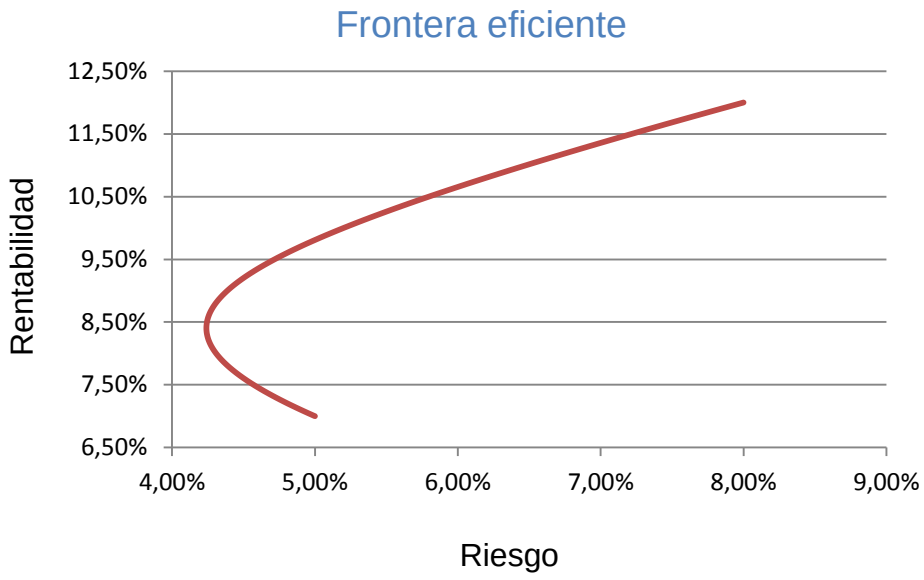

Figura 5.2: Frontera eficiente

5.2. CASO DE ESTUDIO: PROBLEMA DE SELECCIÓN DE CARTERAS DE INVERSIÓN

Es posible considerar muchas variantes del problema de la selección de carteras de inversión, de diferentes niveles de complejidad. El modelo planteado, asume que el objetivo del inversor es construir una cartera que minimice el riesgo para alcanzar una rentabilidad esperada. El problema específico que vamos a tratar presenta las siguientes características generales:

- El 100% del presupuesto tiene que ser invertido en la cartera,
- se limitan las operaciones a crédito, y
- se establecen el número máximo de activos que pueden ser incluidos en la cartera de inversión. Esta condición es interesante por razones de diversificación.

La selección de una cartera en el que desea minimizar el riesgo para alcanzar una rentabilidad esperada requiere la optimización del siguiente problema [CRAM01]:

Minimizar el riesgo de la cartera:

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{ij}x_i x_j = \sigma^2(R_p)$$

Sujeto a:

$$E[R_p] \leq \sum_{i=1}^{n}\overline{b_i}x_i$$

$$\sum_{i=1}^{n}x_i = 1$$

$$0 \leq x_i \leq 1 \quad \forall i = 1, \ldots, n$$

$$\sum_{i=1}^{n}\delta_i \leq K \text{, siendo } \delta_i = \begin{cases} 0 & \text{si } x_i = 0 \\ 1 & \text{si } x_i > 0 \end{cases}$$

donde n es el número de activos que pueden incluirse en la cartera, x_i es el tanto por uno de la cartera invertida en el activo i, σ_i la desviación típica del rendimiento del activo i y σ_{ij} es la covarianza del rendimiento del activo i con el rendimiento del activo j, δ_i es una variable binaria que toma valor 1 cuando $x_i > 0$ y toma valor 0 cuando $x_i = 0$, y K es el número máximo de activos seleccionables para la cartera.

5.2.1. Técnicas para la Optimización de Carteras

El problema de optimización de carteras ha sido tratado lo largo del tiempo por diversas técnicas que, en su mayoría, están basadas en el Modelo de Markowitz. La complejidad para resolver el problema de optimización de carteras se encuentra en el tipo de restricciones que contiene y de los activos a incluir en la cartera.

En el Modelo básico de Markowitz descrito anteriormente, la solución más simple del problema se obtiene cuando se prescinde de la restricción de no-negatividad (posibilidad de venta de activos a corto plazo). Para este planteamiento, el método clásico de Lagrange resulta muy efectivo para encontrar soluciones de manera sencilla. Si incluimos, la restricción de no-negatividad, el problema se vuelve más complejo pudiendo resolverse mediante algoritmos de Programación Cuadrática (adaptación de Wolfe al método Simplex) [WOLF59].

El mundo real financiero requiere determinadas exigencias, restricciones de comercio, número máximo de activos en las carteras, etc.. Estas nuevas premisas conllevan que el problema se convierta en entero mixto no-lineal haciendo que los algoritmos clásicos sean incapaces de resolverlo [CRAM01], [CRAM03]. Diferentes investigadores, en las últimas décadas, han tratado de resolver este tipo de problemas como es el caso de Perold [PERO84] que amplió el número de restricciones del modelo de Markowitz pero sin limitaciones en el número de activos. Si a este planteamiento, se incorporan, restricciones de porcentaje de activo, de comercio, de cardinalidad, la complejidad se acrecienta excesivamente en donde los métodos tradicionales no son eficaces.

La aplicación de algoritmos meta-heurísticos en la optimización de carteras de inversión ha sido tratada por diferentes investigadores. Loraschi, Tettamanzi, Tomassini y Verdarajan [LORA95] propusieron utilizar Algoritmos Evolutivos al igual que Vedarajan, Chan y Goldberg [VEDA97], Bauer y Fitz-Gerald [BAUE00] y Korczak [KORC03] entre otros. Bauer y Fitz-Gerald [BAUE00] demostraron algunos de los potenciales de usar procesos evolutivos en la búsqueda de reglas atractivas de comercio. Chang, Meade, Beasley y Sharaiha [CHAN00] trabajaron con diferentes meta-heurísticas, incluyendo el Recocido Simulado, en un modelo en el que no incluyeron restricciones de comercio. Maringer y Kellerer [MARI03] obtuvieron resultados de alta calidad al problema de optimización de carteras con restricciones de cardinalidad utilizando un algoritmo híbrido de búsqueda local que combina los principios del Recocido Simulado con estrategias evolutivas. Mansini y Speranza [MANS99],

[MANS03], Chiodi [CHIO03], kellerer [KELE00] han realizado diferentes trabajos sobre el problema de optimización de carteras con restricciones de cardinalidad y teniendo en cuenta costes mínimos de transacción aplicando técnicas heurísticas.

Uno de los trabajos de optimización con un elevado número de restricciones – limitaciones de comercio y el número máximo de activos – es el presentado por Crama y Schyns [CRAM01], [CRAM03] en el que utilizan el algoritmo Recocido Simulado para la solución del problema de optimización.

Schlottmann y Seese [SCHL04] realizan una justificación acerca del porqué las aplicaciones financieras se clasifican dentro de los problemas NP y del hecho de que casi todos los problemas con valor práctico y específicamente los de finanzas, caen dentro de la clase de problemas NP-completos, debido a que tienen una estructura combinatoria la cual es equivalente – con respecto a las reducciones en tiempo polinomial – a dichos problemas NP-completos. Por ejemplo, la selección de carteras es equivalente al conocido problema NP-completo del Problema de la Mochila (*Knapsack Problem*).

Estos investigadores consideran que desde el punto de vista de la complejidad computacional, el Modelo planteado por Markowitz es equivalente a resolver un problema semejante al Problema de la Mochila usando variables de decisión con valores reales. En la formulación del Problema de la Mochila se consideran variables de decisión binarias, por lo tanto la complejidad es menor, si bien aquí, la función objetivo no es lineal. Schlottmann y Seese hacen referencia a varias publicaciones donde se ha implementado un Algoritmo Evolutivo con una restricción para la supervisión del presupuesto de capital.

Fieldsend, Matatko y Peng [FIEL04] realizan una implementación del Modelo de Markowitz con restricción de cardinalidad utilizando un algoritmo evolutivo, donde sin nombrar específicamente el estilo de la representación como Problema de la Mochila, utilizan un procedimiento similar de generación de los individuos.

5.3. FORMULACIÓN MATEMÁTICA DEL PROBLEMA DE SELECCIÓN DE CARTERAS DE INVERSIÓN

Para poder determinar la mejor solución al problema de optimización de la Selección de Carteras de Inversión lo vamos a formular en términos matemáticos. Puede definirse en sus aspectos fundamentales de la siguiente forma:

> *El problema de optimización de cartera implica la asignación de inversiones a un número de diferentes activos para minimizar el riesgo para alcanzar una rentabilidad esperada, en un período de inversión dado.*

Como ya hemos comentado anteriormente, las carteras de inversión se componen de un conjunto de activos financieros, cada uno de los cuales tiene un riesgo y una rentabilidad. Estos activos, considerados en conjunto, ofrecen la posibilidad de obtener mejores relaciones rentabilidad/riesgo a las logradas invirtiendo en instrumentos financieros de forma individual.

Para seleccionar óptimamente una cartera de inversión sobre *n* activos, se plantea de la siguiente manera:

- Cada activo es caracterizado por un rendimiento que varía aleatoriamente con el tiempo. El riesgo de cada activo es medido por la variación de su rendimiento. Si cada x_i – *incógnita del problema* – del vector $\boldsymbol{x} = (x_1,\ldots,x_n)$ representa la proporción del presupuesto del inversor destinado al activo financiero *i* en la cartera, entonces el rendimiento total de la cartera está dado por el producto escalar del vector $\boldsymbol{x}$ por el vector de rendimiento de los activos individuales $\boldsymbol{b} = (b_1,\ldots,b_n)$. Cada b_i representa el rendimiento esperado del activo *i*.

- Si tenemos la matriz C_{nxn}, de las covarianzas de los *n* rendimientos, se puede obtener el rendimiento medio (rendimiento esperado de la cartera) por la expresión $\sum_{i=1}^{n} \overline{b_i} x_i$ y su riesgo definido por $\sum_{i=1}^{n}\sum_{j=1}^{n} \sigma_{ij} x_i x_j$

  donde σ_{ij} es la covarianza entre los rendimientos de los valores *i* y *j*

El Problema de optimización de una cartera *p* de inversión (Modelo Markowitz) requiere definir previamente los parámetros del problema:

- Definir el nº de activos (productos financieros) disponibles.
- Calcular los rendimientos de cada activo.
- Calcular la matriz de covarianza C_{nxn}.
- Calcular el rendimiento o rentabilidad de la cartera *p*: $E[R_p] = \sum_{i=1}^{n} \overline{b_i} x_i$
- Calcular el riesgo de la cartera *p*: $\sigma^2(R_p) = \sum_{i=1}^{n}\sum_{j=1}^{n} \sigma_{ij} x_i x_j$

  $\sigma^2(R_p)$ es la varianza de la cartera *p*. $E[R_p]$ es la rentabilidad o rendimiento de la cartera *p*, que son el conjunto de proporciones que minimizan el riesgo de la cartera.
- Definir función objetivo y restricciones. La teoría de Markowitz presenta un modelo simplificado de la realidad. El mundo real financiero requiere determinadas exigencias y limitaciones. Estos factores, que el modelo de Markowitz no contempla, se convierten en restricciones de carácter realista que deben ser tenidas en cuenta por los gestores de cartera. Una de las principales es establecer la cardinalidad de la cartera, es decir, limitar el número de productos distintos que pueden incluirse en la cartera de inversión de modo que sea posible establecer un máximo y mínimo número de activos a

invertir. Conviene establecer la cardinalidad de una cartera por razones de diversificación.

A continuación, se va a realizar una definición matemática formal del problema de la selección de carteras de inversión como problema de optimización combinatoria. Para ello es necesario considerar una discretización de las variables del problema, las cuales representan el tanto por uno seleccionado en la cartera de inversión para cada uno de los activos financieros disponibles. Teóricamente, estas variables toman valores reales en el intervalo [0,1], por lo que en el dominio discreto a considerar se tomarán *m*+1 valores uniformemente distribuidos en dicho intervalo (incluyendo obviamente los valores 0 y 1). El valor usual de *m* empleado en la práctica es *m* = 100, lo que supone como dominio para las variables el conjunto *D*={0,0'01,0'02,0'03,...,0'99,1}.

Definición (Problema de la Selección de Carteras de Inversión). El Problema de la Selección de Carteras de Inversión (*Portfolio Investment Problem, PIP*) consiste en diseñar una cartera *p* que minimice el riesgo para alcanzar una rentabilidad esperada mínima a partir de un conjunto de *n* activos financieros infinitamente divisibles. Formalmente, el problema puede describirse mediante la cuádrupla (*X,S,f,R*):

- $X = \{x_1,..., x_n\}$, es el conjunto de variables del problema, en el que cada variable x_i representa la proporción (tanto por uno) del presupuesto del inversor destinado al activo *i* en la cartera *p*. El dominio finito para cada una de las *n* variables es un conjunto de *m*+1 valores reales uniformemente distribuidos en el intervalo [0,1] (incluyendo los valores 0 y 1).

- $S = D^n$, es el espacio de soluciones del problema, de tamaño $\#S = (m+1)^n$. Una solución arbitraria del problema se representa mediante un vector $\boldsymbol{x} = (x_1,\ldots,x_n) \in S$. La estructura de la solución se codificada en un vector $(x_1,\ldots,x_n)$, donde cada variable x_i representa la

proporción del activo *i* en la cartera *p*. Si $x_i = 0$, el activo *i* no está incluido en la cartera.

- $f: S \to \mathbb{R}^+ \cup \{0\}$, es la función objetivo a minimizar, función determinada por la expresión:

$$f(\boldsymbol{x}) = \sum_{i=1}^{n}\sum_{j=1}^{n} \boldsymbol{\sigma}_{ij} x_i x_j = \boldsymbol{\sigma}^2(R_p)$$

donde σ_{ij} es la covarianza entre los rendimientos de los valores *i* y *j*

- $R = \{r_1, \ldots, r_{n+3}\}$, es el conjunto de restricciones:

$$r_1 : E[R_p] \leq \sum_{i=1}^{n} \overline{b_i} x_i$$

$$r_2 : \sum_{i=1}^{n} x_i = 1$$

$$r_{i+2} : 0 \leq x_i \leq 1 \quad (i = 1, \ldots, n)$$

$$r_{n+3} : \sum_{i=1}^{n} \delta_i \leq K, \quad \delta_i = \begin{cases} 0 & \text{si } x_i = 0 \\ 1 & \text{si } x_i > 0 \end{cases}$$

La restricción r_1 es la restricción de rentabilidad esperada mínima, y expresa la condición de alcanzar la rentabilidad esperada de la inversión; r_2 es la restricción de presupuesto, e indica que el 100% del presupuesto sea invertido en la cartera *p*; r_{i+2}, con $i = 1, \ldots, n$, son las restricciones de no negatividad, e indican que no se permiten ventas a crédito; finalmente, r_{n+3} es la restricción de cardinalidad, que limita a *K* el número máximo de activos distintos que pueden incluirse en la cartera.

El objetivo del problema consiste en encontrar una solución, $\boldsymbol{s}_{\text{opt}} = (s_1, \ldots, s_n) \in S$, que minimice globalmente la función objetivo *f*, de tal forma que se satisfagan todas las restricciones impuestas entre las variables.

El espacio de soluciones viables es el conjunto $S^* = \{\boldsymbol{x} = (x_1,..., x_n) \mid r_1,..., r_{n+3}\} \subseteq S$, formado por las variaciones con repetición de orden *n* de *m*+1 valores discretos en el intervalo [0,1] que suman 1 con al menos *n*–*K* ceros. El tamaño de dicho espacio de soluciones viables no puede determinarse directamente, pero está acotado por $\#S^* < \left(\frac{(n+m-1)!}{m!(n-1)!} \right)$.

Esta definición constituye un modelo matemático que incluye los aspectos principales del problema. Perold enriqueció el Modelo de Markowitz introduciendo ciertas restricciones que reflejan mejor el mundo financiero real [PERO84]. En esta tesis se han planteado algunas de las restricciones propuestas en su Modelo.

El mundo real financiero posee algunas exigencias y requisitos adicionales propios de la selección de carteras de inversión reales que deben tenerse en cuenta por los gestores. Entre las principales restricciones, pueden enumerarse brevemente:

- *Restricciones – Máximo y Mínimo – para la inversión de un activo:* definen los límites superiores e inferiores de la proporción de cada activo permitido incluir en la cartera. Estas limitaciones pueden derivarse de la política de diversificación de la cartera o para descartar inversiones en donde los costes de gestión superen a los posibles beneficios.

- *Restricciones de Volumen de Compra y Venta*: define los límites en la variación de compra o venta de productos financieros respecto a la cartera inicial.

- *Restricciones Mínimas de Comercio*: define la variación mínima de los diferentes activos respecto a la cartera inicial. Para un activo, en un periodo de tiempo, se tienen tres opciones respecto de la cartera inicial – las proporciones del activo permanecen igual o una mínima cantidad debe ser vendida o una mínima cantidad debe ser comprada.

- *Restricción de round-lot* que obliga a que los porcentajes de inversión por activo equivalgan a acciones divisibles por el tamaño del lote.
- *Restricciones de capitalización del mercado* que mide la tendencia a invertir en sectores con mayor valor de capitalización con el objetivo de reducir el riesgo.
- Existen otras restricciones como, por ejemplo, las derivadas de un mercado concreto, de los impuestos, costes de transacciones, etc.

5.4. RESOLUCIÓN DEL PROBLEMA DE LA CARTERA DE VALORES MEDIANTE RECOCIDO SIMULADO CON LA ESTRATEGIA DE ENFRIAMIENTO ADAPTATIVO ÓPTIMO

A continuación, se presentan las pruebas realizadas en la resolución del problema de la cartera de valores mediante la meta-heurística del recocido simulado. Respecto al programa de enfriamiento, se ha utilizado la combinación óptima de parámetros determinada en el capítulo anterior: La temperatura inicial corresponde al 60% del número de ciclos de enfriamiento apropiado para la temperatura inicial estándar, y el parámetro α de reducción de la temperatura varía con porcentaje de incremento del 3%, con lo que se mantiene un objetivo medio y una desviación típica del objetivo estadísticamente similares a los del algoritmo estándar, reduciendo al máximo el tiempo de ejecución. Se han realizado un total de 100 ejecuciones tomando como base la misma instancia del problema de la cartera de valores, obteniéndose la media, y la desviación típica del objetivo y del número de iteraciones. La instancia utilizada en las pruebas es una muestra de 46 activos del índice EURO STOXX 50 sobre un periodo total de 12 años. El software desarrollado para soportar el algoritmo del recocido simulado con la estrategia de enfriamiento adaptativo y su aplicación al problema de selección de carteras de inversión, se ha realizado mediante una metodología de diseño y programación orientada a objetos en el lenguaje de programación C++. Las

pruebas se han ejecutado en un computador personal con unos tiempos que oscilan entre 17 y 49 segundos para el algoritmo con enfriamiento adaptativo óptimo, y entre 82 y 117 segundos para el algoritmo con enfriamiento estándar. La proporción entre el número total de iteraciones y el tiempo de ejecución es constante en ambos casos, aproximadamente igual a 35000 iteraciones por segundo. Por ello, en la tabla de resultados se proporciona como información de convergencia solo la relativa al número total de iteraciones efectuadas.

La tabla 5.1 muestra los valores promedios de rentabilidades y desviaciones típicas semanales calculados en base a datos semanales de 12 años para una muestra de 46 activos del Índice EURO STOXX 50, utilizados en la aplicación del problema de selección de carteras de inversión.

Activos	Acrónimo	Rentabilidad media	Desviación típica de la rentabilidad
AIR LIQUIDE SA	AI	0,17%	3,35%
ALLIANZ SE-REG	ALV	-0,04%	5,44%
ALSTOM	ALO	0,01%	8,24%
ARCELORMITTAL	MT	0,42%	7,99%
AXA SA	CS	0,06%	6,12%
BANCO SANTANDER SA	SAN	0,09%	4,89%
BASF SE	BAS	0,23%	4,38%
BAYER AG-REG	BAYN	0,15%	4,82%
BAYERISCHE MOTOREN WERKE AG	BMW	0,22%	4,71%
BANCO BILBAO VIZCAYA ARGENTA	BBVA	0,05%	4,84%
BNP PARIBAS	BNP	0,19%	5,56%
CARREFOUR SA	CA	-0,08%	4,16%
CRH PLC	CRH	0,10%	4,85%
DAIMLER AG-REGISTERED SHARES	DAI	0,08%	5,21%
DANONE	BN	0,14%	3,36%
DEUTSCHE BANK AG-REGISTERED	DBK	0,08%	5,94%
DEUTSCHE TELEKOM AG-REG	DTE	-0,22%	4,88%
E.ON SE	EOAN	0,16%	4,09%
ENEL SPA	ENEL	-0,02%	3,34%
ENI SPA	ENI	0,15%	3,82%
FRANCE TELECOM SA	FTE	-0,15%	6,24%
ASSICURAZIONI GENERALI	G	-0,04%	3,79%
IBERDROLA SA	IBE	0,17%	3,66%
ING GROEP NV-CVA	INGA	0,07%	6,88%
INTESA SANPAOLO	ISP	0,06%	5,27%

KONINKLIJKE PHILIPS NV	PHIA	0,09%	5,65%
L'OREAL	OR	0,07%	3,40%
LVMH MOET HENNESSY LOUIS VUI	MC	0,15%	4,66%
MUENCHENER RUECKVER AG-REG	MUV2	-0,01%	4,87%
NOKIA OYJ	NOK1V	-0,11%	6,18%
REPSOL SA	REP	0,08%	3,97%
RWE AG	RWE	0,13%	3,94%
COMPAGNIE DE SAINT-GOBAIN	SGO	0,14%	5,22%
SANOFI	SAN	0,11%	4,06%
SAP AG	SAP	0,19%	6,06%
SCHNEIDER ELECTRIC SA	SU	0,18%	4,71%
SIEMENS AG-REG	SIE	0,16%	5,31%
SOCIETE GENERALE	GLE	0,17%	5,90%
TELECOM ITALIA SPA	TIT	-0,15%	5,09%
TELEFONICA SA	TEF	0,07%	4,13%
TOTAL SA	FP	0,11%	3,61%
UNIBAIL-RODAMCO SE	UL	0,35%	3,56%
UNICREDIT SPA	UCG	0,02%	5,53%
UNILEVER NV-CVA	UNA	0,09%	3,45%
VINCI SA	DG	0,30%	3,93%
VIVENDI	VIV	-0,09%	5,61%

Tabla 5.1: Promedios de rentabilidades y desviaciones típicas semanales
Fuente: Elaboración propia a partir de Bloomberg.

A continuación, se muestran los parámetros del problema de la selección de carteras de inversión que se han utilizado para las pruebas:

- Número de activos n = 46
- Número máximo de activos en la cartera K = 15
- Número de valores de discretización m = 100
- Rentabilidad mínima 0.002

La tabla 5.2, muestra los parámetros de los programas de enfriamiento estándar y enfriamiento adaptativo óptimo utilizados en el *problema de selección de carteras de inversión*:

Enfriamiento estándar	Enfriamiento adaptativo óptimo
- Temperatura inicial (100%) $T_0 = 0{,}0007580744$ - Alfa 0.95 - Nº de iteraciones para cada ciclo de temperatura: 13917 - Criterio de fin (nº de iteraciones sin mejoras) 21393	- Temperatura inicial (60%) $T_0 = 0{,}0000128155$ - Alfa inicial 0.8 - Nº de iteraciones para cada ciclo de temperatura: 13917 - Factor de incremento de alfa 0.03 - Criterio de fin (nº de iteraciones sin mejoras) 21393

Tabla 5.2: Parámetros de los programas de enfriamiento

En la tabla 5.3, están recogidos los datos de número de iteraciones, temperatura inicial, temperatura final y objetivo, junto con la media, desviación, valor máximo y valor mínimo, para cada una de las 100 ejecuciones con los dos programas de enfriamiento – enfriamiento multiplicativo exponencial estándar y enfriamiento adaptativo óptimo – tomando como base la misma instancia del problema de la selección de cartera de valores.

Enfriamiento multiplicativo exponencial estándar				
	Iteraciones	T_0	T_n	Objetivo (%)
Media	2934367,17	0,0007580744	0,0000000188	0,0582%
Desviación	197662,64	0,0000000000	0,0000000098	0,0005%
Máximo	4078132	0,0007580744	0,0000000467	0,0602%
Mínimo	2631928	0,0007580744	0,0000000002	0,0576%
Enfriamiento adaptativo óptimo				
	Iteraciones	T_0	T_n	Objetivo (%)
Media	1022164,15	0,0000128155	0,0000000207	0,0582%
Desviación	210026,14	0,0000000000	0,0000000091	0,0007%
Máximo	1727303	0,0000128155	0,0000000625	0,0627%
Mínimo	616783	0,0000128155	0,0000000062	0,0576%

Tabla 5.3: Resultados programas de enfriamiento estándar y adaptativo óptimo

Cartera óptima

En las figuras 5.3 y 5.4, se muestran las carteras óptimas de inversión formadas por los diferentes activos y el porcentaje de la inversión destinado a dicho valor con el programa de enfriamiento estándar y con el programa de enfriamiento adaptativo óptimo.

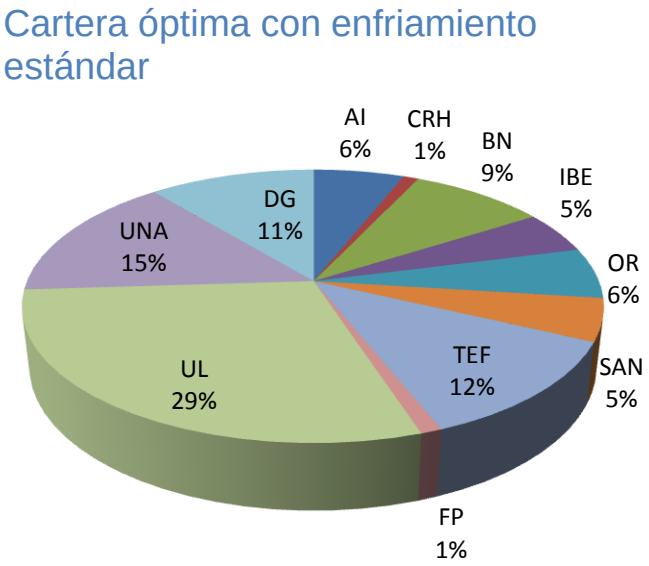

Figura 5.3: Cartera de inversión con el programa de enfriamiento estándar (solución óptima)

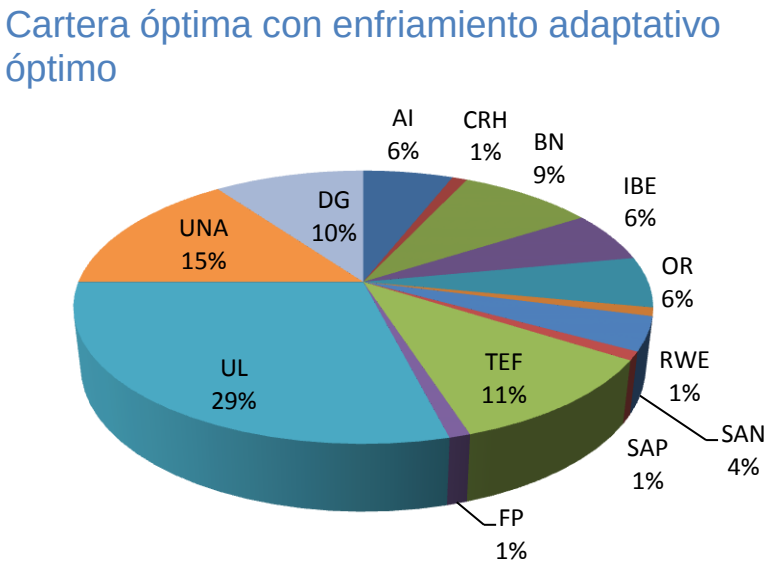

Figura 5.4: Cartera de inversión con el programa de enfriamiento adaptativo óptimo (solución óptima)

Capítulo 6: Conclusiones y Líneas Futuras

"No hay secretos para el éxito. Este se alcanza preparándose, trabajando arduamente y aprendiendo del fracaso"

Colin Powell

6.1. CONCLUSIONES SOBRE EL TRABAJO REALIZADO

El objetivo principal de esta tesis doctoral, enmarcada dentro del área de la optimización combinatoria, ha sido el desarrollo de una estrategia de enfriamiento adaptativo para el algoritmo del recocido simulado.

Con este fin, se ha realizado un trabajo de investigación que ha obtenido los siguientes resultados:

- Se ha realizado una revisión general del campo de la optimización combinatoria, de los problemas fundamentales que surgen en este campo, y de los principales algoritmos meta-heurísticos de resolución. Para ello, en primer lugar se han expuesto las definiciones, terminología y notación de los conceptos fundamentales de la optimización combinatoria: *definición formal del problema genérico de optimización combinatoria, soluciones y sus tipos, complejidad algorítmica de los problemas de optimización combinatoria*. A continuación, se ha realizado una descripción formal de los problemas fundamentales de optimización combinatoria: *el problema del viajante comercial, el problema de la planificación de rutas de distribución, el problema del empaquetado, el problema de la partición, el problema de la asignación cuadrática, el problema de la mochila, el problema de la máxima diversidad y el problema de la programación de operaciones de producción*. Estos

problemas han servido como referencia para el estudio de nuevas técnicas de optimización combinatoria durante las últimas décadas, por lo que se consideran problemas fundamentales de la optimización combinatoria. Finalmente, dado que la presente tesis se centra en el análisis detallado y la mejora de una de las técnicas de resolución más relevantes, el recocido simulado, y con el fin de comprender mejor y enmarcar este algoritmo, se ha realizado una breve descripción de las técnicas meta-heurísticas más importantes que pueden encontrarse en la literatura científica: *búsqueda tabú, algoritmos evolutivos y genéticos, optimización por enjambre de partículas inteligentes, búsqueda dispersa, algoritmo GRASP, optimización por colonia de hormigas,* etc., estableciéndose una completa taxonomía o clasificación básica de las meta-heurísticas de acuerdo con diversos criterios.

- Para el estudio del algoritmo meta-heurístico del recocido simulado, se han presentado los fundamentos físicos en que está basado y se ha descrito detalladamente este algoritmo. Con el objetivo de diseñar una estrategia de enfriamiento adaptativo para el algoritmo del recocido simulado, se ha realizado *un primer estudio comparativo riguroso entre las principales funciones de reducción de la temperatura aplicables al programa de enfriamiento característico del recocido simulado.* Para ello se ha evaluado un conjunto representativo de programas con diferentes funciones de enfriamiento de la literatura: cuatro de enfriamiento monótono multiplicativo, cuatro de enfriamiento monótono aditivo, y uno de enfriamiento no monótono adaptativo [DIAZ08]. Todos ellos se componen al menos de los tres parámetros planteados en el programa de enfriamiento original de Kirkpatrick, Gelatt y Vecchi [KIRK83]: temperatura inicial T_0, función de decremento de la temperatura (característica objeto de la comparativa), y número L de transiciones de estado para cada valor de la temperatura. Precisamente, como conclusión de este estudio puede afirmarse que el enfriamiento multiplicativo exponencial, el estándar propuesto por Kirkpatrick, Gelatt y Vecchi, es uno de los mejores, sin diferencias estadísticamente

significativas en eficiencia respecto a las restantes mejores funciones de enfriamiento contrastadas.

- El recocido simulado se caracteriza por sus propiedades de convergencia hacia soluciones de alta calidad pero con un elevado tiempo computacional. Con esta premisa, esta investigación se ha centrado en reducir el tiempo de ejecución del algoritmo manteniendo la calidad de las soluciones. Teniendo esto en cuenta, se han realizado tres estudios originales sobre la influencia de los parámetros del programa de enfriamiento en el comportamiento del algoritmo, utilizando como referencia el enfriamiento multiplicativo exponencial, y ejecutando las pruebas sobre dos casos concretos (instancias) de dos problemas clásicos de optimización combinatoria: el problema del viajante comercial (TSP), y el problema de la asignación cuadrática (QAP). Estos estudios son: *la influencia de la temperatura inicial en el comportamiento del algoritmo*; *la influencia del parámetro* α *cuando éste se modifica dinámicamente en la ecuación de reducción exponencial de la temperatura, y la influencia combinada de la temperatura inicial y el parámetro* α *en la eficiencia del algoritmo.*

- Finalmente, se ha desarrollado *una aplicación práctica sobre una importante área de gran interés en la economía: El problema de Optimización de Carteras de Inversión*. Se ha realizado una definición matemática formal del problema de la selección de carteras de inversión como problema de optimización combinatoria. Se ha implementado una aplicación informática que resuelve dicho problema mediante la meta-heurística del recocido simulado, y se ha realizado un conjunto de pruebas utilizando en el programa de enfriamiento la combinación óptima de parámetros determinada como consecuencia de los tres estudios previos.

Los resultados de los estudios anteriormente citados permiten extraer las siguientes conclusiones generales:

- En el *estudio comparativo entre funciones de decremento de la temperatura aplicables al programa de enfriamiento del algoritmo meta-heurístico del recocido simulado,* con respecto al nivel de la calidad del objetivo y su relación con la forma de la curva de decrecimiento de la temperatura, puede afirmarse que el algoritmo del recocido simulado funciona bien respecto a su capacidad de escape de óptimos locales cuando la curva tiene pendiente moderada en las partes inicial y central del procesamiento, y más suave en la parte final del mismo, tal como sucede en el enfriamiento multiplicativo exponencial estándar. La forma concreta (convexa, sigmoidal) de la curva en las partes inicial y central no parece relevante, pero en la parte final una curva de descenso más suave puede favorecer mejores niveles de optimización. Con respecto al tiempo de ejecución, todos los programas de enfriamiento estudiados para el recocido simulado presentan un coste similar, debido –lógicamente – a que sus parámetros han sido configurados para asemejarse en este comportamiento al enfriamiento multiplicativo exponencial de referencia. Sin embargo, la desviación típica del número de iteraciones efectuadas parece estar en relación con la cola de la curva de reducción de la temperatura en la fase final del algoritmo. Una cola inversamente logarítmica produce un descenso final de la temperatura más suave y de mayor desviación, mientras que las colas de tipo inversamente lineal o cuadrático generan menor desviación. Precisamente el enfriamiento multiplicativo exponencial y el enfriamiento adaptativo, basado en el anterior, poseen una cola que se anula más rápidamente y proporcionan los mejores valores de desviación temporal del algoritmo.

- La conclusión del estudio de la influencia de la temperatura inicial en la eficiencia del recocido simulado es que para un porcentaje de ciclos de enfriamiento igual o superior al 50% la calidad de la solución media obtenida (objetivo) no es significativamente diferente (peor) que la obtenida con la temperatura inicial estándar.

- La conclusión del estudio de la influencia de la variación dinámica del parámetro α en la eficiencia del recocido simulado es que el enfriamiento con un porcentaje de variación dinámica del parámetro α del 2% proporciona un objetivo estadísticamente similar al enfriamiento estándar, con reducción del tiempo de ejecución (iteraciones).

- La conclusión del estudio combinado de la influencia del parámetro α dinámico y la temperatura inicial en la eficiencia del recocido simulado es que el algoritmo con enfriamiento adaptativo basado en la variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 60% proporciona un objetivo medio y una desviación típica del objetivo estadísticamente similares a los del enfriamiento estándar, reduciendo el tiempo de ejecución (iteraciones).

- La principal conclusión del análisis conjunto de los tres estudios realizados es que el algoritmo con enfriamiento adaptativo basado en la variación dinámica del parámetro α del 3% en combinación con un porcentaje del número de ciclos de temperatura del 60% mantiene un objetivo medio y una desviación típica del objetivo estadísticamente similares a los del algoritmo estándar, reduciendo al máximo el tiempo de ejecución (iteraciones) respecto a todos los casos contemplados (enfriamiento adaptativo óptimo). De hecho, la reducción del tiempo de ejecución es muy notable: el enfriamiento adaptativo óptimo requiere el 30% del tiempo medio que emplea el enfriamiento estándar con los valores usuales de los parámetros.

- Finalmente, el buen comportamiento presentado por la estrategia de enfriamiento adaptativo del algoritmo del recocido simulado en la implementación del problema de optimización de carteras de inversión, tanto en la calidad de las soluciones obtenidas como en el tiempo medio de ejecución, ha demostrado la adecuación de dicha estrategia para la resolución de problemas de optimización combinatoria de interés práctico.

6.2. LÍNEAS FUTURAS DE INVESTIGACIÓN

Las líneas futuras de investigación que pueden desarrollarse como extensión del trabajo realizado resultan en su mayor parte evidentes a partir de la lectura de esta memoria.

Las líneas de investigación básica se centran en los siguientes aspectos:

- En la tesis se han realizado diversos estudios dirigidos a determinar la influencia de determinados parámetros del programa de enfriamiento en la eficiencia de recocido simulado, concretamente la función de reducción de la temperatura en un primer estudio básico, y en los estudios originales posteriores la temperatura inicial y la variación dinámica del parámetro α para el enfriamiento multiplicativo exponencial, más la influencia combinada de ambos. Sin embargo, no se ha realizado ninguna comprobación de la influencia del tercer parámetro que aparece en el programa de enfriamiento original de Kirkpatrick, Gelatt y Vecchi: el número *L* de transiciones de estado en cada ciclo de temperatura. De acuerdo con el fundamento físico teórico del recocido simulado, este parámetro debe asegurar con probabilidad próxima a la unidad que en cada valor de la temperatura se alcanza el equilibrio térmico, lo que trasladado al ámbito de los problemas de optimización combinatoria supone que en cada temperatura debe poderse explorar completamente la vecindad de la solución en curso. Teniendo en cuenta esto, el valor de *L* se ha calculado siempre a partir del tamaño *m* de la vecindad de las soluciones como el número de transformaciones necesarias para explorar la vecindad de la solución en curso con una probabilidad de 0,95. Resultaría interesante conocer la influencia de este parámetro *L* en la eficiencia del recocido simulado con diferentes probabilidades de exploración de la vecindad.

- Estudiar nuevos métodos de reducción adaptativa de la temperatura, combinados con el enfriamiento multiplicativo exponencial. Estos métodos podrían basarse, como el de Locatelli utilizado en la comparativa realizada en el capítulo 2 de esta tesis, en modificar la

temperatura en función de la distancia del objetivo en curso a un mejor objetivo de referencia. Este objetivo de referencia podría ser el mejor objetivo conseguido hasta el momento, una cota inferior del óptimo global, etc.

- Estudiar la utilidad del enfriamiento adaptativo óptimo en otras meta-heurísticas trayectoriales y poblacionales que han sido combinadas con el recocido simulado, tales como la búsqueda tabú con recocido simulado, o los algoritmos genéticos y evolutivos estocásticos.

Por otra parte, las líneas de investigación aplicada a desarrollar son:

- Desarrollar un modelo del problema de optimización de carteras de inversión que incorpore todas las características del mundo real financiero, es decir, que tenga en cuenta las restricciones realistas asociadas.

- Finalmente, estudiar la utilidad de la estrategia de enfriamiento adaptativo en la implementación de otros problemas prácticos de optimización combinatoria tales como el problema de la asignación de rutas a vehículos en el ámbito del transporte y la distribución, el problema de la planificación de turnos de trabajo en el ámbito laboral, el problema de la programación de producción en la organización industrial, etc.

Bibliografía

BIBLIOGRAFÍA

[AART89] Aarts, E.H.L. & Korst, J.: *Simulated Annealing and Boltmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*. Wiley, Chichester, 1989.

[AART97] Aarts, E.H.L. & Lenstra, J.: *Local Search in Combinatorial Optimisation*. Wiley, Chichester, 1997.

[ACKL87] Ackley, D.H.:. *A Connectionist Machine for Genetic Hillclimbing*. Kluwer, 1987.

[ADED08] Adedoyin, O.: *Application of Heuristic Methods To Portfolio Optimisation: An Object-Oriented Approach*. University of Nottingham, 2008.

[AHN02] Ahn, C.W. & Ramakrishna, R.S.: A genetic algorithm for shortest path routing problem and the sizing of populations. IEEE Transactions on Evolutionary Computation, Vol. 6, No. 6, pp. 566–579, 2002.

[AKBA10] Akbari R., Mohammadi M., Ziarati K., “A novel bee swarm optimization algorithm for numerical function optimization”, In: Journal of Communications in Nonlinear Science and Numerical Simulation 15(2010) 3142-3155.

[ALBA99] Alba, E.: Análisis y Diseño de Algoritmos Genéticos Paralelos Distribuidos. PhD thesis, Universidad de Málaga, 1999.

[ALBA02] Alba E., Tomassini M., Parallelism and Evolutionary Algorithms, IEEE Transactions on Evolutionary Computation, IEEE Press, 6(5):443-462, October 2002.

[ALBA03] Alba E., Luque G., Algoritmos Híbridos y Paralelos para la Resolución de Problemas Combinatorios, Dpto. Informática-Univ. Oviedo (ed.), Actas del Segundo Congreso Español de Meta-heurísticas y Algoritmos Evolutivos y Bioinspirados (MAEB’03), Gijón, pp. 353-362, 2003.

[ALBA03] Alba, E., Laguna, M. y Martí, R. "Métodos evolutivos en problemas de optimización". REVISTA INGENIERÍA UC. Vol. 10, No 3, 80-89, 2003.

[ALBR06] Albrecht, A.: *A Stopping Criterion for Logarithmic Simulated Annealing.* Computing 78, 55-79. Springer – Verlag 2006.

[ALVA03] Álvarez, E., Díaz, F., Ruiz, E., de los Santos, A. & López, B. (2003): *A Genetic Algorithm for Real-Time Job-Shop Scheduling*. International Conference on Flexible Automation & Intelligent Manufacturing, FAIM-2003, 81-92.

[AMIR07] K.S. Amirthagadeswaran & V.P. Arunachalam: Enhancement of Performance of Genetic Algorithm for Job Shop Scheduling Problems through Inversion Operator. International Journal of Advanced Manufacturing Technology, 32 (2007), 780-786.

[APOL89] Apolloni, B., Caravalho, N., Falco de, D.: *Quantum stochastic optimization,* Stochastic Proceses their Applications, 33, 233-244 (1989).

[ARTU06] Arturo, E., Hernan, J., Moreno, A.: *Negociación de portafolios de acciones usando la meta-heurística Recocido Simulado*. Scientia et Technica Año XII, No 30, Mayo de 2006 UTP.

[BAKE87] Baker J. E.: Adaptive selection methods for genetic algorithms. In J. Grefenstette, editor, 2nd International Conference On Genetic Algorithms, pages 100_11. 1987.

[BAKE03] Baker, B.M. & Ayechew, M.A.: A genetic algorithm for the vehicle routing problem. Computers & Operations Research, Vol. 30, No. 5, pp. 787–800, 2003.

[BANH10] Banharnsakun, A., Achalakul, T. & Sirinaovakul, B.: Artificial bee colony algorithm on distributed environments. In IEEE Second World Congress on Nature and Biologically Inspired Computing, pp. 13–18, 2010.

[BARA02] Barán, B. (2002), Parallel Asynchronous Team Algorithms, In: Correa, R., Dutra, I., Fiallos, M.: Models for Parallel and Distributed Computation: Theory, Algorithmic Techniques and Applications, Kluwer Academic Publishers.

[BAUE00] Bauer, R. Fitz-Gerald, F.G.: Using genetic programming to design a generalized trading system. Managerial Finance, 26 (6), 1-15. 2000.

[BEIE02] Beielstein, T., Parsopoulos, K. E. and Vrahatis, M.N.: “Tuning PSO parameters through sensitivity analysis”. Technical Report of the Collaborative Research Center, University of Dortmund, 2002.

[BELM13] Belmecheri, F., Prins, C., Yalaoui, F. & Amodeo, L.: Particle swarm optimization algorithm for a vehicle routing problem with heterogeneous fleet, mixed backhauls, and time windows. Journal of intelligent manufacturing, Vol. 24, No. 4, pp. 775–789, 2013.

[BENI96] Benítez, D., Ramos, R. (1996), Partición de Sistemas de lineales para su resolución en Sistemas Distribuidos, XXII Conferencia Latino - Americana de Informática (PANEL 96), Bogotá-Colombia.

[BERR99] R. Berretta and P. Moscato. The number partitioning problem: An open challenge for evolutionary computation ? In D. Corne, M. Dorigo, and F. Glover, editors, New Ideas in Optimization, pages 261–278. McGraw-Hill, Maidenhead, Berkshire, England, UK, 1999.

[BERR03] Berretta, R. Cotta, C. and Moscato, P.: Enhancing the performance of memetic algorithms by using a matching-based recombination algorithm: Results on the number partitioning problem. In M. Resende and J. Pinho de Sousa, editors, Metaheuristics: Computer-Decision Making, pages 65–90. Kluwer Academic Publishers, Boston MA, 2003.

[BEZA08] Bezákova, I., Stefankovic, D., Vazirani, V., Vigoda, E. (2008): *Accelerating simulated annealing for the permanent and combinatorial counting problems*. SIAM J. COMPUT, Volo. 37, No. 5, pp 1429-1454.

[BIAN93] Bianchini R., Brown C. (1993). “Parallel Genetic Algorithms on Distributed-Memory Architectures”. Technical Report 436 (revised version), May.

[BLUE03] Blue C., Roli A., Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison, ACM Computing Surveys 35(3), 268-308, 2003.

[BOER04] Boeringer, D.W., Werner, D.H.: “Particle swarm optimization versus Genetic algorithms for phased array synthesis”, *IEEE Transactions on Antennas and Propagation*, Vol. 52, No. 3, March 2004, pp. 771-779.

[BOLU11] Bolufé Röhler and Stephen Chen, “An Analysis of Sub-swarms in Multi-swarm Systems”, in Lecture Notes in Artificial Intelligence, Vol. 7106: Proceedings of Joint Australasian Conference in Artificial Intelligence, D. Wang and M. Reynolds Eds. Springer-Verlag 2011, pp. 271–280.

[BONA99] Bonabeau, E., Dorigo, M. and Theraulaz, T.: *From Natural to Articial Swarm Intelligence*. Oxford University Press, 1999.

[BONA99] Bonabeau, E., Dorigo, M. y Théraulaz, G. (1999). Swarm Intelligence: From Natural to Artificial Systems. New York: Oxford University Press.

[BONA99] Bonabeau, E. y Meyer, C. (2001). Swarm intelligence: A whole new way to think about business. Harvard Business Review, 79(5), 107–114.

[BRAS97] Brassard, Gilles; Bratley, Paul (1997). «Algoritmos voraces». Fundamentos de Algoritmia. Madrid: PRENTICE HALL.

[BRAT08] Bratton, D.; Blackwell, T. (2008). «A Simplified Recombinant PSO». Journal of Artificial Evolution and Applications.

[BUCK92] Buckles, B.P. and Petry, F.E.: *Genetic Algorithms*. IEEE Computer Society Press, 1992.

[BULL97] Bullnheimer, B., Hartl, R. F., Strauss, C., A new rank-based version of the ant system: a computational study., Technical Report POM-03/97, Institute of Management Science, University of Vienna, 1997.

[CALG99] Calgari, P.R.: Parallelization of Population - Based Evolutionary logarithms for Combinatorial Optimization Problems. Doctoral thesis. Éole Polytechnique Fédérale de Lausanne. Lausanne - Francia, 1999.

[CAMP92] Campello R., Maculan N. Algoritmos e Heurísticas Desenvolvimiento e avaliaçao de performance. Apolo Nacional Editores, Brasil (1992).

[CANT98] Cantú-Paz, E.: A survey of parallel genetic algorithms. Calculateurs Paralleles, Réseaux et Systems Répartis, 10(2):141–171, 1998.

[CANT98] Cantú-Paz, E.: A survey of parallel genetic algorithms. Calculateurs paralleles, reseaux et systems repartis, Vol. 10, No. 2, pp. 141–171, 1998.

[CARL01] Carlisle, A. (2001). «An Off-The-Shelf PSO». Proceedings of the Particle Swarm Optimization Workshop: 1–6.

[CARR02] Carreto, C., Baker, B.: *A Grasp Interactive Aproach to the Vehicle Routing Problem with Backhauls* pp. 185-199. En Eassys and Surveys in Metaheuristics. Celso C. Ribero, Pierre Hansen (Eds) Kluwer Academic Publishers, 2002.

[CARR92] Carrizo, F.G. Tinetti, and P. Moscato. A computational ecology for the quadratic assignment problem. In Proceedings of the 21st Meeting on Informatics and Operations Research, Buenos Aires, Argentina, 1992. SADIO.

[CERN85] Cerny, V.: *Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm*. Journal of Optimization Theory and Applications, 45:41-51, 1985.

[CHAN98] Chand S; Schneeberger H. Single machine scheduling to minimize earliness subject to notardy jobs. European Journal of Operational Research, 34: 221-230 (1988).

[CHAN00] Chang, T. J., Meade, N., Beasley, J.E. and Sharaia, Y. M. “Heuristics for Cardinality Constrained Portfolio Optimization”, Computers and Operations Research, vol. 27, 1271-1302, 2000.

[CHAN05] Chan, F.T.S., Wong, T.C. & Chan, L.Y.: A Genetic Algorithm-Based Approach to Machine Assignment Problem. International Journal of Production Research vol. 43, no. 12 (2005), 2451-2472.

[CHAN06] F.T.S. Chan, T.C. Wong & L.Y. Chan: Flexible Job-Shop Scheduling Problem under Resource Constraints. International Journal of Production Research vol. 44, no. 11 (2006), 2071-2089.

[CHANJ05] Chanj, J.F., Chu, S., Roddick, J.F. & Pan, J.: A Parallel particle swarm optimization algorithm with communication strategies. Journal of Information Science and Engineering, Vol. 21, No. 4, pp. 809–818, 2005.

[CHEN11] Chen, S. and Montgomery, J.:“Selection Strategies for Initial Positions and Initial Velocities in Multi–optima Particle Swarms”, in Proceedings of the Genetic and Evolutionary Computation Conference, 2011 pp. 53–60.

[CHIO03] Chiodi, Mansini, Speranza: Semi-Absolute Deviation Rule for Mutual Funds Portfolio Selection. Annals of Operations Research, 124(1-4), 245-265.

[CHIP96] Chipperfield A., Fleming P. (1996). “Parallel Genetic Algorithms”. Parallel and Distributed Computing Handbook, Zomaya A. Y. H. (ed.), MacGraw-Hill, 1118-1143.

[CHUB97] P.C. Chu and J. Beasley. A genetic algorithm for the generalised assignment problem. Computers & Operations Research, 24:17–23, 1997.

[CLER02] Clerc, M.; Kennedy, J. (2002). «The particle swarm - explosion, stability, and convergence in a multidimensional complex space». IEEE Transactions on Evolutionary Computation 6 (1): pp. 58–73.

[COLO91] Colorni, A., Dorigo, M., Maniezzo, V., Distributed optimization by ant colonies., Memorias del European Conference on Artificial Life (ECAL91), Elsevier Publishing, 1991, pp. 134–142.

[COLO92] Colorni, A., Dorigo, M., Maniezzo, V., An investigation of some p roperties of an “ant algorithm”., Memorias del Parallel Problem Solving from Nature Conference, Elsevier Publishing, 1992, pp. 509–520.

[CORD99] Cordón, F. Herrera, L. Moreno. Integración de Conceptos de Computación Evolutiva en un Nuevo Modelo de Colonias de Hormigas. VIII Conferencia de la Asociación Española para la Inteligencia Artificial, (Seminario Especializado en Computación Evolutiva), Murcia (España), 1999, Vol. II, páginas 98-105.

[CORD00] Cordón, O., de Viana, I. F., Herrera, F., Moreno, L., A new aco model integrating evolutionary computation concepts: The best-worst ant system. Memorias del ANTS 2000 - From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms, 2000, pp. 22–29.

[CORM01] Cormen T., Leiserson Ch. And Rivest R.: Introduction to Algorithms, MIT Press, Editorial McGraw Hill (2001).

[COST95] Costa, N. Dubuis, and A. Hertz. Embedding of a sequential procedure within an evolutionary algorithm for coloring problems in graphs. Journal of Heuristics, 1(1):105–128, 1995.

[COTT07] Cotta C. 2007. “Una Visión General de los Algoritmos Meméticos”. 1ª Edición. Procedimientos Metaheurísticos en Economía y Empresa. España, ASEPUMA. p. 139-166.

[COTT03] Cotta, C., Moscato, P.: A gentle introduction to memetic algorithms. In G. Kochenberger F. Glover, editor, *Handbook of Metaheuristics*, volume 105-144. Kluwer Academic Publishers, 2003.

[CRAM01] Crama Y., Schyns, “Simulated Annealing for Complex Portfolio Selection Problems”, 2001.

[CRAM03] Crama Y., Schyns, “Simulated Annealing for Complex Portfolio Selection Problems”, European Journal of Operational Research, 150 (3), 546-571. 2003.

[DASC05] Das, A. and Chakrabarti, B.K. (Eds.): *Quantum Annealing and Related Optimization Methods,* Lecture Note in Physics, Vol. 679, Springer, Heidelberg 2005.

[DAVI85] Davis L.: Applying adaptive algorithms to domains. Proceedings of the International Joint Conference on Artificial Intelligence, pages 162_164, 1985.

[DAVI85] Davis, L.: Job shop scheduling with genetic algorithms. In Proceedings of the 1st international conference on genetic algorithms, pp. 136–140. L. Erlbaum Associates Inc., 1985.

[DAVI91] Davis, L.D.: Handbook of Genetic Algorithms. Van Nostrand Reinhold Computer Library,New York (1991).

[DAWK76] Dawkins, R.: The Selfish Gene. Clarendon Press, Oxford, UK, 1976.

[DIAZ96] Díaz, J.F.: Optimización Combinatoria. En ODE: Un Nuevo Modelo Neuronal Estocástico para Optimización Combinatoria. Tesis Doctoral, pp 13-30. Universidad de Deusto, 1996.

[DIAZ96] Díaz, A., Glover, F., Ghaziri, H.M., Gonzalez, J.L., Laguna, M, Moscato, P. y Tseng, F.T.: *Optimización Heurística y Redes Neuronales*, Paraninfo, Madrid. 1996.

[DIAZ03] Díaz, F., Álvarez, E., Ruiz, E., de los Santos, A. & López, B.: *Algoritmo Evolutivo Heurístico para Programación de Producción Discreta Orientada al Proceso*. II Congreso Español de Meta-heurísticas y Algoritmos Evolutivos y Bioinspirados, MAEB 2003, 562-569. 2003.

[DIAZ08] Díaz, F., Riaño, J.: *A Comparison of Cooling Schedules for Simulated Annealing.* Encyclopedia of Artificial Intelligence.

[DIAZ11] Díaz, J.F.: Algoritmo Evolutivo para Programación de Producción Discreta Orientada al Proceso: Un Estudio Comparativo de Convergencia. Proyecto PRORRECO: Modelo Integrado de Programación de la Producción en Redes de Fabricación Colaborativas, 2011.

[DOWS01] Doewsland, K.A. & Díaz, A.: *Diseño de Heurísticas y Fundamentos del Recocido Simulado*. Revista Iberoamericana de Inteligencia Artificial, 20: 34-52, 2001.

[DOWS03] Dowsland, K., Adenso Díaz, B.: *Diseño de heurística y fundamentos del recocido simulado*. Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial, año/vol. 7, número 019, 2003.

[DORI91] Dorigo, M., Maniezzo, V., Colorni, A., Ant system: An autocatalytic optimizing process, Technical Report 91-016, Politecnico di Milano, Italy, 1991a.

[DORI91] Dorigo, M., Maniezzo, V., Colorni, A., Positive feedback as a search strategy, Technical Report 91-016, Politecnico di Milano, Italy, 1991b.

[DORI96] Dorigo, M., Maniezzo, V., Colorni, A.: “The ant system: Optimization by a colony of cooperating agents”, *IEEE Transactions on Systems, Man and Cybernetics, Part B*, Vol. 26, No. 1, 1996, pp. 29-41.

[DORI97] Dorigo, M. and Gambardella, L.M.: Ant Colony System: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, 1(1):53-66, 1997.

[DORI99] Dorigo, M. and Di Caro, G.: The Ant Colony Optimization meta-heuristic. En D. Corne, M. Dorigo, y F. Glover, editores, *New Ideas in Optimization*, páginas 11-32. McGraw Hill,London, UK, 1999.

[DORI03] Dorigo, M. and Stutzle, T.: The ant colony optimization meta-heuristic: Algorithms applications, and advances. In F. Glover and G. Kochenberger, editors, *Handbook on MetaHeuristics*. 2003.

[DUAR07] Duarte Muñoz, A.: Meta-heurísticas. Editorial Dykinson. 2007. Pág 72-95.

[EBER00] Eberhart, R.C. (2000). «Comparing inertia weights and constriction factors in particle swarm optimization». Proceedings of the Congress on Evolutionary Computation Vol. 1: 84–88.

[EL-AB10] El-Abd, M.: A cooperative approach to the artificial bee colony algorithm. In IEEE Congress on Evolutionary Computation, pp. 1–5, 2010.

[FERN05] Fernández, J., Barán, B. (2005): Equipo Elitista de Algoritmos Evolutivos Multiobjetivo. XXXI Conferencia Latinoamericana de Informática – CLEI-2005. Cali – Colombia.

[FEOR89] Feo, T.A., Resende, M.G.C.: *A Probabilistic Heuristic for a Computationally Difficult Set Covering Problem* Operations Research Letters 8 pp. 67-71, 1989.

[FEOR95] Feo, T.A., Resende, M.G.C.: *Greedy Randomized Adaptive Search Procedures* Journal of Global Optimization 6 pp. 109-133. 1995.

[FEST02] Festa, P. & Resende, M.G.C.: GRASP: An annotated bibliography. In C.C. Ribeiro and P. Hansen, editors, Essays and Surveys on Metaheuristics, pp. 325–367, Kluwer Academic Publishers, 2002.

[FIEL04] Fieldsend, J., Matatko, J. and Peng, M.: Cardinality constrained portfolio optimisation. In Proceedings of the Fifth International Conference on Intelligent Data Engineering and Automated Learning (IDEAL'04), volume Lecture Notes in Computer Science (LNCS 3177), pages 788.793. Z.R. Yang, R. Everson and H. Yin (Eds.), Springer, August 25-27 2004.

[FOGE96] Fogel, L.,, Owens, A. and Walsh, M.: Arti_cial Intelligence Thorough Simulated Evolution. John Wiley, Chichester, UK, 1996.

[FRAN00] Franklin, B. and Bergerman, M.: Cultural algorithms: Concepts and experiments. In *Proc. of the 2000 Congress on Evolutionary Computation*, pages 1245–1251, Piscataway, NJ, 2000. IEEE Service Center.

[GARB95] Gambardella, L. M., Dorigo, M., Ant-Q: A reinforcement learning approach to the traveling salesman problem, Memorias del Twelfth International Conference on Machine Learning, Morgan Kaufmann, 1995, pp. 252–260.

[GARE79] Garey, M.R. & Johnson, D.S.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, 1979.

[GARI12] Garitselov, O., Mohanty, S. P. and Kougianos, E.: “Accurate Polynomial Metamodeling-Based Ultra-Fast Bee Colony Optimization of a Nano-CMOS PLL”- Special Issue on Power, Parasitics, and Process-Variation (P3) Awareness in Mixed-Signal Design, Journal of Low Power Electronics (JOLPE), Volume 8, Issue 3, June, 2012, pp. 317--328.

[GEMA84] Geman, S. & Geman, D.: *Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images*. IEEE Transactions of Pattern Analysis and Machine Intelligence, 6: 721-741, 1984.

[GLOV86] Glover, F.: *Future Paths for Integer Programming and Links to Artificial Intelligence*. Computers and Operations Research, 5: 533-549, 1986.

[GLOV89] Glover, F.: Tabu search. part I. *ORSA Journal on Computing*, 1:190-206, 1989.

[GLOV90] Glover, F.: Tabu search. part II. *ORSA Journal on Computing*, 2:4-32, 1990.

[GLOV93] Glover, F. and De Werra, D.: *Tabu Search*, volume 41. Baltzer, 1993.

[GLOV97] Glover, F. and Laguna, M.: *Tabu Search*. Kluwer, 1997.

[GLOV93] Glover, F., De Werra, D. and Taillard, E.: A user's guide to tabu search. *Annals of Operations Research*, 41:3-28, 1993.

[GLOV93] Glover, F.; Laguna, M. (1993). Tabu Search. In REEVES, C. (Ed.) Modern Heuristic Techniques for Combinatorial Problems. Blackwell Scientific Publishing, Oxford, pp. 70-141.

[GLOV97] Glover, F. and Laguna, M. (1997). Tabu Search, Kluwer Academic Publishers.

[GLOV98] Glover F. (1998). “A Template for Scatter Search and Path Relinking. J.k. Hao, E. Lutton, E. Ronald, M. Schoenauer, D. Syners (eds.). Artificial Evolution. Springer LNCS 1363, pp. 13-64.

[GLOV99] Glover, F. (1999). Scatter Search and Path Relinking. In CORNE, D.; DORIGO, M.; GLOVER, F. (Eds.) New Methods in Optimization. McGraw Hill.

[GLOV00] Glover, F.: Multi-start and strategic oscillation methods - principles to exploit adaptive memory. In M.Laguna and J.L. González-Velarde, editors, *Computing Tools for Modeling, Optimization and Simulation*, pages 1-24. Kluwer Academic Publishers, 2000.

[GLOV03] Glover, F. and Kochenberger, G. editors. *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2003.

[GLOV04] Glover, F. , Laguna, M., Martí, R., New Ideas and Applications of Scatter Search and Path Relinking, to appear in New Optimization Techniques in Engineering, Springer-Verlag, G. Onwubolu y B.V.Babu (Eds.) (2004).

[GLOV06] Glover, F., Laguna, M. and Martí R. (2006) Principles of Tabu Search. To appear in Approximation Algorithms and Metaheuristics, Chapman & Hall/CRC.

[GOLD95] Goldberg D. E., et al. (1995). “Critical Deme Size for Serial and Parallel Genetic Algorithms”.IlliGAL Report No. 95002.

[GOLD89] Goldberg, D.E.: *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA, 1989.

[GOME00] Gómez, B.F.: *Gestión de Cartera*. Descleé de Brouser. 2000.

[GUTI02] Gutin, G., Yeo, A. and Zverovich, A.: Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the TSP. Discrete Applied Mathematics 117 (2002), 81–86.

[HADI10] Hadidi, A. Sina Kazemzadeh Azad, Saeid Kazemzadeh Azad, Structural optimization using artificial bee colony algorithm, 2nd International Conference on Engineering Optimization, 2010, September 6 – 9, Lisbon, Portugal.

[HAJE88] Hajek, B.: *Cooling Schedules for Optimal Annealing*. Mathematics of Operations Research, 13: 311-329, 1988.

[HANS01] Hansen P. and N. Mladenovic. Variable neighborhood search: Principles and applications. European Journal of Operational Research, 130:449-467, 2001.

[HANS02] Hansen P. and N. Mladenovic. Variable neighborhood search. In P.M. Pardalos and M.G.C. Resende, editors, Handbook of Applied Optimization, pages 221-234. Oxford University Press, 2002.

[HANS03] Hansen P. and N. Mladenovic. Variable neighborhood search. In F. Glover and G.A. Kochenberger, editors, Handbook of Metaheuristics, chapter 6. Kluwer Academic, 2003.

[HANS10] Hansen P., Mladenovic N., Moreno J.A. Variable Neighborhood Search: methods and applications. Annals of Operations Research, 175(1):367–407 (2010).

[HASE04] Hasegawa, M.: *A Concept of Effectively Global Search in Optimization by Local Search Heuristics*. SLOW DYNAMICS IN COMPLEX SYSTEMS: 3rd International Symposium on Slow Dynamics in Complex Systems. AIP Conference Proceedings, Volume 708, pp. 747-748 (2004).

[HART91] Hart, W.E., Belew, R.K.: Optimizing an arbitrary function is hard for the genetic algorithm. In Belew, R.K., Booker, L.B., eds.: Proceedings of the 4th. International Conferenceon Genetic Algorithms, San Mateo CA, Morgan Kaufmann (1991) 190–195.

[HEDA02] Hedar, A.R., Fukushima, M.: Hybrid Simulated Annealing and Direct Search Method for Nonlinear Unconstrained Global Optimization. Optimization Methods and Software, Volume 17, Issue 5 2002 , pages 891 – 912.

[HERN07] Hernan Restrepo Correa, J., Arturo Cruz Trejos, E., Medina Varela, P.D.: *Negociación de portafolios de acciones usando un hibrido de las meta-heurísticas búsqueda dispersa, recocido simulado, y búsqueda tabú.* Revista Facultad de Ciencias Económicas: Investigación y Reflexión, vol XV, núm 2, diciembre 2007, pp. 79-95.

[HOLL62] Holland, J.H.: Concerning efficient adaptive systems. In M. C. Yovitis, G. T. Jacobi, and G. D. Goldstein, editors, *Self-Organizing Systems*, pages 215–230. Spartan Books,Washington, D.C., 1962.

[HOLL73] Holland, J.H.: *Genetic Algorithms and the Optimal Allocation of Trials*. SIAM Journal of Computing, 2: 88-105, 1973.

[HOLL75] Holland, J. H.: Adaptation in Natural and Arti_cial Systems. The MIT Press, Cambridge, Massachusetts, first edition, 1975.

[HOLS99] Holstein, D. and Moscato, P.: Memetic algorithms using guided local search: A case study. In D. Corne, M. Dorigo, and F. Glover, editors, New Ideas in Optimization, pages 235–244. McGrawHill, Maidenhead, Berkshire, England, UK, 1999.

[HOOS05] Hoos, H.H.; Stützle, T. Stochastic local search: foundations and applications, Morgan Kaufmann Publishers, San Francisco, CA, 2005.

[JOHN85] Johnson, D.S., Papadimitriou, C.H. & Yannakakis, M.: *How Easy is Local Search?*. Proceedings of the Annual Symposium on Foundations of Computer Science, 39-42. Los Angeles, CA, 1985.

[JOHN95] Johnson, J, Rahmat-Samii, Y.: “Genetic algorithm optimization of wireless communication networks”, *Proceedings of the IEEE Antennas and Propagation Society International Symposium*, California (USA), June 1995, Vol. 4, pp. 1964-1967.

[KARA05] Karaboga, D.: An Idea Based On Honey Bee Swarm for Numerical Optimization, Technical Report-TR06,Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[KARA07] Karaboga, D. and Basturk, B:. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. Journal of Global Optimization, 39 (3):459–471, 2007.

[KARA08] Karaboga, D. and Basturk, B "On the performance of artificial bee colony (ABC) algorithm", in Journal of Applied Soft Computing, Vol. 8, pp. 687–697, 2008.

[KARA09] Karaboga D., Akay B., "A comparative study of Artificial Bee Colony algorithm", in Journal of Applied Mathematics and Computation, 2009.

[KARA12] Karaboga, D., Gorkemli, B., Ozturk, C. & Karaboga, N.: A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artificial Intelligence Review, pp. 1–37, 2012.

[KARP72] Karp, R.: Reducibility among combinatorial problems. In R. Miller & J. Thatcher (eds.): Complexity of Computer Computations. Plenum Press, pp 85-103, 1972.

[KELL00] Kellerer, Mansini y Speranza: Selecting portfolios with fixed costs and minimum transaction lots. Annals of Operations Research (99), 287-304. 2000.

[KENN95] Kennedy, J., Eberhart, R.C.: "Particle swarm optimization", *Proceedings of the IEEE International Conference on Neural Networks-ICNN'95*, Perth (Australia), December 1995, Vol.4, pp. 1942-1948.

[KENE95] Kennedy, J., Eberhart, R.C.: A new optimizer using particle swarm theory. Proceedings of the Sixth International Symposium on Micromachine and Human Sciencie, Nagoya, Japan, pp 39-43.1995

[KENN01] Kennedy J., Eberhart, R.C.: *Swarm Intelligence*. Morgan Kaufmann, 2001.

[KIRK83] Kirkpatrick, S., Gelatt, C.D. & Vecchi, M.P.: *Optimization by Simulating Annealing*. Science, 220: 671-680, 1983.

[KOCH03] Kochenberger G. A.: Handbook of Metaheuristics. Boston/Dordrecht/London, Kluwer Academic Publishers, University of Colorado at Denver, 2003.

[KORC03] Korczak, J. J., Lipinsky, P. And Roger, P. “Evolution Strategy in Portfolio Optimization”, Proceedings of the 5th. International Conference on Artificical Evolution. Lecture Notes in Computer Science, 2310, 156-167, Springer 2003.

[KOZA90] Koza J. R.: Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems. Technical Report STAN-CS-90-1314, Stanford University, 1990.

[KRAS04] Krasnogor, N. and Gustafson, S.: “A study on the use of self generation" in memetic algorithms”. *Natural Computing*, 3: 53-76, 2004

[LAAR87] Laarhoven, P.J.M. Van & Aarts, E.H.L.: *Simulated Annealing: Theory and Applications*. Reidel, Dordrecht, 1987.

[LAGU91] Laguna, M., Barnes, J. W. and Glover, F.: Tabú Search Methods for a Single Machine Scheduling Problem, Journal of Intelligent Manufacturing, vol 2, 63-74. 1991.

[LAGU94] Laguna, M.; Feo, T.; Elrod, H. (1994). A Greedy Randomized Adaptative Search Procedures for the 2-Partition Problem. Operations Research, 42(4):677-687.

[LAGU00] Laguna, M, F. Glover, and R. Martí. Control and Cybernetics, 39:653-684, 2000.

[LAGU02] Laguna, M and Martí, R.. The optquest callable library. In Stefan Voss and David L. Woodru, editors, Optimization Software Class Libraries, pages 193{218. Kluwer Academic Publishers, 2002.

[LAGU03] Laguna M. Martí R., Scatter Search: Methodology and Implementations in C, Kluwer Academic Publishers. 2003.

[LAGU03] Laguna, M, F. Glover, and R. Martí. In F. Glover and G. Kochen-berger, editors, Handbook on MetaHeuristics, chapter 1. Scatter search and path relinking: Advances and applications, 2003.

[LINK73] Lin, S.; Kernighan, B.W.. An Effective Heuristic Algorithm for the TravelingSalesman Problem, Operations Research, 1973, Vol. 21, p. 498-516.

[LOCA00] Locatelli, M.: *Convergence of a Simulated Annealing Algorithm for Continuous Global Optimization*. Journal of Global Optimization, 18: 219-234, 2000.

[LOCA01] Locatelli, M.: *Convergence and first hitting time of simulated annealing algorithms for continuous global optimization.* Mathematical methods of operations research. ISSN 1432-2994. Vol. 54, n^{o}2, pp. 171-199 (27 ref.). 2001.

[LORA95] Loraschi, Tettamanzi, Tomassini y Verda: Distributed genetic algorithms with an application to portfolio selection problems. Proc. Int. Conf. On Artificial Neural Networks an d Genetic Algorithms, 384-387. 1995.

[LOUR01] Lourenço, H.R.; Martin, O.C.; Stüzle, T. (2001). Iterated Local Search. In GLOVER, F.; KOCHENBERGER, G. (Eds.) Handbook of Metaheuristics. Kluwer Academic Publishers.

[LUKE05] Luke, B.T.: *Simulated Annealing*. Technical Report. http://fconyx.ncifcrt.gov/~lukeb/simanfl.html, 2005.

[MAND89] Manderick B. & Spiessens, P.: Fine-grained parallel genetic algorithms. In Proceedings of the third international conference on Genetic algorithms, pp. 428–433. Morgan Kaufmann Publishers Inc., 1989.

[MANS99] Mansini, Speranza: Heuristic algorithms for the portfolio selection problema with mínimum transaction lots. European Journal of Operational Research (114), 219-233. 1999.

[MANS03] Mansini, Ogryczak, Speranza: Lp solvable models for portfolio optimization: A classification and computational comparison. Journal of Management Mathematic. 2003.

[MARK52] Markowitz, H.: *Portfolio Selection*. In: The Journal of Finance Vol. 7, No. 1: pp. 77-91. March, 1952.

[MARK70] H. Markowitz. Portfolio selection: efficient diversification of investments. Blackwell, New York, 2nd edition, 1991, John Wiley & Sons, 1959, Yale University Press, 1970.

[MARI03] Maringer, D., Kellerer, H. Optimization of Cardinality constrained portfolios with a hybrid local search algorithm. Or spectrum, 25 (4), 481-495. 2003.

[MARI05] Maringer Diezmar, “Portfolio Management with Heuristic Optimization”, Springer, 2005.

[MARI13] Marinakis, Y., Iordanidou, G. & Marinaki, M.: Particle swarm optimization for the vehicle routing problem with stochastic demands. Applied Soft Computing, Vol. 13, No. 4, pp. 1693–1704, 2013.

[MART03] Martí, R.: Multistart methods. In Fred Glover and Gary A. Kochenberger, editors, *Handbook of Metaheuristics*, pages 355-368. Kluwer Academic, 2003.

[MART03] Martí, R. and M. Laguna. Scatter search: Diseño básico and estrategias avanzadas. Inteligencia Artifcial. Revista Iberoamericana de Inteligencia Artifcial, 19:123{130, 2003.

[MART10] Martínez F.J., González-Vidosa F., Hospitaler A. and Yepes V. (2010) "Heuristic optimization of RC bridge piers with rectangular hollow sections". Computers & Structures, 88:375-386.

[MASC12] Mascareñas, J.: *Gestión de carteras I: Selección de carteras.* Universidad Complutense de Madrid. Última versión: diciembre 2012.

[MATT04] Dirk C. Mattfeld & Christian Bierwirth: An Efficient Genetic Algorithm for Job-Shop Scheduling with Tardiness Objectives. European Journal of Operational Research 155 (2004), 616-630.

[MERK02] Merkle, D., Middendorf, M. and Schmeck, H.: “Ant Colony Optimization for Resource – Constrained Project Scheduling”, IEEE Trans on Evolutionary Computing, Vol 6 No 4, pp333-346, Aug 2002.

[MERZ99] P. Merz and B. Freisleben. A Comparison of Memetic Algorithms, Tabu Search, and Ant Colonies for the Quadratic Assignment Problem. In

Pete Angeline, editor, 1999 Congress on Evolutionary Computation (CEC'99), pages 2063–2070, Piscataway, NJ, USA, 1999. IEEE Press.

[MERZ00] Merz P. and Freisleben B.: Fitness Landscapes, Memetic Algorithms and Greedy Operators for Graph Bi-Partitioning.Evolutionary Computation, 8(1):61–91, 2000.

[METR53] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. & Teller, E.: *Equation of State Calculations by Fast Computing Machines*. Journal of Chemical Physics, 21:1087-1092, 1953.

[MEUL00] Meuleau, N., Dorigo, M., Ant colony optimization and stochastic gradient descent, Technical Report TR/IRIDIA/ 2000-36, IRIDIA, Universit´e Libre de Bruxelles, Belgium, 2000.

[MICH92] Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag, 1992.

[MICH00] Michalewicz, Z and Fogel, D.B.: *How to Solve It: Modern Heuristics*. Springer Verlag, 2000.

[MITC78] Mitchell, T.: *Version Spaces: An Approach to Concept Learning*. PhD thesis, Computer Science Department, Stanford University, Stanford, California, 1978.

[MITC96] Mitchel, M.: *An introduction to Genetic Algorithms*. MIT Press, 1996.

[MITR86] Mitra, D., Romeo, F. & Sangiovanni-Vincentelli, A.L.: *Convergence and Finite-time Behavior of Simulated Annealing*. Advances in Applied Probability, 18: 747-771, 1986.

[MLAD97] Mladenovic, N. and P. Hansen. Variable neighborhood search.Com. Oper. Res, 24:1097 – 1100, 1997.

[MOON08] Ilkyeong Moon, Sanghyup Lee & Hyerim Bae: Genetic Algorithms for Job-Shop Scheduling Problems with Alternative Routings. International Journal of Production Research vol. 46 no. 10 (2008), 2695-2705.

[MOSC89] Moscato, P.: "On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms". *Caltech Concurrent Computation Program*, C3P Report 826, 1989.

[MOSC92] Moscato and M. G. Norman. A Memetic Approach for the Traveling Salesman Problem Implementation of a Computational Ecology for Combinatorial Optimization on Message-Passing Systems. In M. Valero, E. Onate, M. Jane, J. L. Larriba, and B. Suarez, editors, Parallel Computing and Transputer Applications, pages 177–186, Amsterdam, 1992. IOS Press.

[MOSC99] Moscato, P.: *Memetic Algorithms: A Short Introduction*. In D. Corne, M. Dorigo & F. Glover (eds.), *New Ideas in Optimization*, 219-234. McGraw-Hill, Maidenhead, Berkshire, England, UK, 1999.

[MOSC03] Moscato, P. and Cotta, C.: Una introducción a los algoritmos meméticos. *Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial*, 19:131-148, 2003.

[MOSC04] Moscato, P., Cotta, C., Mendes, A.S.: Memetic algorithms. In Onwubolu, G.C., Babu,B.V., eds.: New Optimization Techniques in Engineering. Springer-Verlag, Berlin Heidel-berg (2004) 53–85.

[NIU07] Niu, B., Zhu, Y., He, X. & Wu, H.: MCPSO: A multi-swarm cooperative particle swarm optimizer. Applied Mathematics and Computation, Vol. 185, No. 2, pp. 1050–1062, 2007.

[OSAB13] Osaba, E., Onieva, E., Carballedo, R., Díaz, F., Perallos, A. & Zhang, X.: A multi-crossover and adaptive island based population algorithm for solving routing problems. Journal of Zhejiang University SCIENCE C, Vol. 14, No. 11, pp. 815–821, 2013.

[OSMA96] OSMAN, I.H.; KELLY, J.P. (Eds.) (1996). Meta-Heuristics: Theory & Applications. Kluwer Academic Publishers.

[OSTE08] Ostertag A., Doerner K F, Hartl R F, Taillard E D, and Waelti P, “POPMUSIC for a real-world large-scale vehicle routing problem with time windows,” Journal of the Operational Research Society, no. 7. pp. 934–943, 2008.

[PARP02] Parpinnelli, R., Lopes, H. and Freitas, A.: “Data Mining with an Ant Colony Optimization Algorithm”, IEEE Trans on Evolutionary Computing, Vol 6 No 4, pp321-332, Aug 2002.

[PARP10] Parpinelli, R.S., Vargas, C.M. & Lopes, H.S.: Parallel approaches for the artificial bee colony algorithm. In Handbook of Swarm Intelligence, pp. 329–345. Springer, 2010.

[PERO84] Perold, A. F.: Large-scale portfolio optimisation. Management Science (30) 1143-1160. 1984.

[PEZZ08] F. Pezzella, G. Morganti & G. Ciaschetti: A Genetic Algorithm for the Flexible Job-Shop Scheduling Problem. Computers & Operations Research 35 (2008), 3202-3212.

[PIRL96] Pirlot, M.: General local search methods. *European Journal of Operational Research*, 92(3):493-511, 1996.

[POLI07] Poli, R. (2007). Department of Computer Science, University of Essex, UK. ed. «An analysis of publications on particle swarm optimisation applications». Technical Report CSM-469.

[POLI08] Poli, R. (2008). «Analysis of the publications on the applications of particle swarm optimisation». Journal of Artificial Evolution and Applications 2008: pp. 1-10.

[QIAN08] Qian, Bin; Wang, Ling; Huang, De-Xian & Wang, Xiong: Scheduling Multi-Objective Job Shops Using a Memetic Algorithm Based on Differential Evolution. The International Journal of Advanced Manufacturing Technology, vol. 35 no. 9-10 (2008): pp1014-1027.

[RAHM99] Rahmat-Samii, Y., Michielssen, E.: “Electromagnetic optimization by genetic algorithms”, John Wiley & Sons, New York, 1999.

[RAYW95] Rayward-Smith, V.J.: *Applications of Modern Heuristic Methods*. Alfred Waller, 1995.

[RECH73] Rechenberg, I. Evolutionsstrategie: Optimierung technischer Systemenach Prinzipien der biologischen Evolution. Frommann-Holzbog, Stuttgart, 1973. German.

[REEV93] Reeves, C.R.: Modern heuristic techniques for combinatorial problems. John Wiley & Sons, Inc., 1993.

[REEV95] Reeves, C.R.: *Modern Heuristic Techniques for Combinatorial Problems*, McGraw-Hill, UK. 1995.

[REEV96] Reeves, C.R.. Hybrid genetic algorithms for bin-packing and related problems. Annals of Operations Research, 63:371–396, 1996.

[REEV02] Reeves, C.R. and Rowe, J.E., editors. Genetic Algorithms-Principles and Perspectives. A Guide to GA Theory, volume 20 of Interfaces in Computer Science and Operations Research. Kluwer Academic Publishers, 2002.

[REEV03] Reeves, C.R.: Genetic algorithms. In F. Glover and G. Kochen-berger, editors, *Handbook on MetaHeuristics*, chapter 3. 2003.

[REIL97] Reilly, F. and Brown, C. K.: 1997, Investment Analysis and Portfolio Management, The Dryden Press.

[RENF94] Renfrew, A. C. Dynamic Modeling in Archaeology: What, When, and Where? In S. E. van der Leeuw, editor, Dynamical Modeling and the Study of Change in Archaelogy. Edinburgh University Press, Edinburgh, Scotland, 1994.

[RESE00] Resende, Mauricio G.C. *GRASP: Greedy Randomized Adaptative Search Procedure. A methaheuristic for combinatorial optimization*. Nueva Jersey, 2000.

[RESE03] Resende, M.G.C and Ribeiro, C.C.: Greedy randomized adaptive search procedures. In F. Glover and G.G. Kochenberger, editors, *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2003.

[REYN94] Reynolds, R.: An Introduction to Cultural Algorithms. In A. V. Sebald and L. J. Fogel, editors, Proceedings of the Third Annual Conference on Evolutionary Programming, pages 131–139.World Scientific, River Edge, New Jersey, 1994.

[REYN05] Reynolds, R. Bin Peng. Knowledge Learning and Social Swarms in Cultural Systems. Journal of Mathematical Sociology. 29:1-18, 2005.

[REYN07] Reynolds, R. and Ali, M Z., Exploring Knowledge and Population Swarms via an Agent-Based Cultural Algorithms Simulation Toolkit (CAT), in proceedings of IEEE Congress on Computational Intelligence 2007.

[REYN08] Reynolds, R. and Ali, M. Z, “Embedding a Social Fabric Component into Cultural Algorithms Toolkit for an Enhanced Knowledge-Driven Engineering Optimization”, International Journal of Intelligent Computing and Cybernetics (IJICC), Vol. 1, No 4, pp. 356–378, 2008

[ROBI04] Robinson, J., Rahmat-Samii, Y.: “Particle swarm optimization in electromagnetics”, *IEEE Transactions on Antennas and Propagation*, Vol. 52, No. 2, February 2004, pp. 397-407.

[ROCC09] Rocca, P.; Benedetti, M.; Donelli, M.; Franceschini, D.; Massa, A. (2009). «Evolutionary optimization as applied to inverse scattering problems». Inverse Problems25: pp. 1–41.

[RODR08] Rodriguez-Tello, E., Hao, J.H., Torres-Jimenez, J.: An effective two-stage simulated annealing algorithm for the minimum linear arrangement problem. Computers and Operations Research. Volume 35. Issue 10 (October 2008).

[RUIZ05] Ruiz, R.; Maroto, C. A comprehensive review and evaluation of permutation flowshop heuristics, European Journal of Operational Research, 2005, Vol. 165 (2), p.479-494.

[SCHL04] Schlottmann, F. and Seese, D.: Financial Applications of Multi-objective Evolutionary Algorithms: recent developments and future research directions. Coello-Coello, C.; Lamont, G. (eds.), World Scientic Singapore, 2004. ISBN 981-256-106-4.

[SCHW94] Schwefel H.: Evolution and Optimum Seeking. John Wiley & Sons, New York, 1994.

[SENT06] P. Senthilkumar & P. Shahabudeen: GA Based Heuristic for the Open Job Shop Scheduling Problem. International Journal of Advanced Manufacturing Technology, 30 (2006), 297-301.

[SHIY98] Shi, Y.; Eberhart, R. (1998). «A modified particle swarm optimizer». Proceedings of IEEE International Conference on Evolutionary Computation: 69–73.

[SHIY98] Shi, Y. and Eberhart, R.: “Parameter Selection in Particle Swarm Optimization”. In Proceedings of the Seventh Annual Conference on Evolutionary Programming, pp. 591-601, 1998.

[SNYM05] Snyman, J.A.: *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms.* Springer Publishing. ISBN 0-387-24348-8, 2005.

[SOLN02] Solnon, C.: “Ants can solve Constrint Satisfaction Problems”, IEEE Trans on Evolutionary Computing, Vol 6 No 4, pp347-357, Aug 2002.

[STEI02] Steinbach Marc C.: *Markowitz Revisited: Mean-Variance Models in Financial Portfolio Analysis.* Disponible: www.fiquam.polytechnique.fr/gt2002/mv.pdf

[STEI05] Stein, M., Branke, J. and Schmeck, H.: *Portfolio Selection: How to Integrate Complex Constraints*. Institute for Applied Informatics and Formal Description Methods, University of Karlsruhe (TH), June 1, 2005.

[STUT99] Stutzle, T. Local search algorithms for combinatorial problems-analysis, algorithms and news applications. DISKI Dissertationen zurKÄunstlikenIntelligenz., 1999.

[STUT00] Stutzle, T. and Hoos, H.: MAX-MIN Ant System. Future Generation Computer Systems, 16(8):889-914, 2000.

[STUT02] Stutzle, T. and Dorigo, M.: “A short convergence proof for a Class of Ant Colony Optimization Algorithm,” IEEE Trans on Evolutionary Computing, Vol 6 No 4, pp 358-365, Aug 2002

[STUT03] Stutzle, T., Martin, O. and Lourenco, H.R.: Iterated local search. In F. Glover and G.G. Kochenberger, editors, Handbook of Metaheuristics, chapter 11. Kluwer Academic Publishers, 2003.

[STUT03] Stutzle, T. Iterated Local Search & Variable Neighborhood Search, M.N. Summerschool, Tenerife, 2003.

[TAIL01] Taillard, E. D. and Voss, S. POPMUSIC -Partial optimization metaheuristic under special intensification conditions, chapter in Essays

and surveys in metaheuristics, pages 613–630. KluwerAcademicPublishers, 2001.

[TAIL02] Taillard, E. D. and Voss, S. (2002). POPMUSIC *Partial Optimization* Metaheuris- tic *under special Intensification Conditions*. In C.C. *Ribeiro* and P. Hansen, Editors, Essays and Surveys in *Metaheuristics*, 613-629.

[TALU83] Talukdar S.N., Pyo S.S., and Giras T.: Asynchronous procedures for parallel processing. IEEE Transaction on PAS, 102(11), 1983.

[TALU97] Talukdar, S., Baerentzen, L., Gove, A., de Souza, P., (1997), Asynchronous Teams: Cooperation Schemes for Autonomous Agents, Journal of Heuristics.

[TALU03] Talukdar S.N., Murthy S., and Akkiraju R.: Asynchronous Teams, chapter 19 in Handbook of Metaheuristic, pages 437–556. Kluwer Academic Publishers, 2003.

[TANE89] R. (1989). “Distributed Genetic Algorithms”. In [Scha89], 434-439.

[TODA83] Toda, M., Kubo, R. & Saitô, N.: *Statistical Physics*. Springer-Verlag, Berlin, 1983.

[TREL03] Trelea, I.C. (2003). «The Particle Swarm Optimization Algorithm: convergence analysis and parameter selection». Information Processing Letters 85 (6): pp. 317–325.

[TSAI09] Tsai, P., Pan, J., Liao, B. & Chu, S.: Enhanced artificial bee colony optimization. International Journal of Innovative Computing, Information and Control, Vol. 5, No. 12, pp. 5081–5092, 2009.

[TUPI01] Tupia, M.: Un algoritmo Voraz Adaptativo y Randómico para resolver el problema de la Programación de Tareas independientes en máquinas homogéneas. Pontificia Universidad Católica del Perú (2001).

[VAND01] Van den Bergh, F. (2001). University of Pretoria, Faculty of Natural and Agricultural Science, ed. An Analysis of Particle Swarm Optimizers (PhD thesis).

[VANH97] Van Horne, J.C.: *Administración Financiera.* Dec. Edic. Prentice Hall Hispanoamericano,S.A, pag. 5.1997.

[VEDA97] Vedarajan, G., Chan, L. C. y Goldberg, D. E.: “Investment Portfolio Optimization Using Genetic Algorithms” , Late Breaking Papers at the Genetic Programming Conference, Koza, J. R. Ed., 255-263, Stanford Bookstore, Stanford Univ., 1997.

[VERC06] Vercher, E.: Optimization in Finance: Portfolio selection models. Departament d’Estadística i Investigació Operativa. Universitat de València. Iberian Conference in Optimization. Coimbra 2006.

[WATA05] Masato Watanabe, Kenichi Ida & Mitsuo Gen: A Genetic Algorithm with Modified Crossover Operator and Search Area Adaptation for the Job-Shop Scheduling Problem. Computers and Industrial Engineering 48 (2005), 743-752.

[WHIT99] Whitley, D., Rana, S. & Heckendorn, R.B.:. The island model genetic algorithm: On separability, population size and convergence. Journal of Computing and Information Technology, Vol. 7, pp. 33–48, 1999.

[WOLF59] Wolfe, P.: The simplex method for quadratic programming. Econometrica (27), 382398. 1959.

[XIAO08] Xiao, L. & Wu, Z.: *Optimization of Security Investment Portfolio Based on Improved Simulated Annealing Algorithm*. School of Management, Tianjin Polytechnic University, Tianjin 300384, China. Vol 3. No 11. 2008.

[XU08] Xu, Y., Wang, Q. & Hu, J.: An improved discrete particle swarm optimization based on cooperative swarms. In IEEE International Conference on Web Intelligence and Intelligent Agent Technology, volume 2, pp. 79–82, 2008.

[YANG09a] Yang, X.: Firefly algorithms for multimodal optimization. In Stochastic algorithms: foundations and applications, pp. 169–178. Springer, 2009.

[YANG09b] Yang, X. & Deb, S.: Cuckoo search via Lévy flights. In World Congress on Nature & Biologically Inspired Computing, pp. 210–214. IEEE, 2009.

[YANG10] Yang, X.: A new metaheuristic bat-inspired algorithm. In Nature inspired cooperative strategies for optimization, pp. 65–74. Springer, 2010.

[ZEIG12] Zeighami V., Akbari R., and Ziarati K. An Efficient Multi Population Artificial Bee Colony. International Journal of Machine Learning and Computing, Vol. 2, No. 3, June 2012.

[ZHAN11] Zhang, Y.& Wu L.: Face Pose Estimation by Chaotic Artificial Bee Colony, International Journal of Digital Content Technology and its Applications, vol. 5, no. 2, (2011), pp. 55-63.