PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

PRUEBA Y COMPARACIÓN DE MÉTODOS DE OPTIMIZACIÓN VÍA SIMULACIÓN BAJO CONDICIONES REALES

MARÍA JOSÉ PÉREZ VILLA

Tesis presentada a la Dirección de Investigación y Postgrado
como parte de los requisitos para optar al grado de
Magister en Ciencias de la Ingeniería

Profesor Supervisor:
PEDRO GAZMURI SCHLEYER

Santiago de Chile, Enero 2010

PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

PRUEBA Y COMPARACIÓN DE MÉTODOS DE OPTIMIZACIÓN VÍA SIMULACIÓN BAJO CONDICIONES REALES

MARÍA JOSÉ PÉREZ VILLA

Miembros del Comité:
PEDRO GAZMURI SCHLEYER
JOSÉ PEDRO PRINA PACHECO
JOSÉ MANUEL ROBLES VÁZQUEZ
RAFAEL RIDDELL CARVAJAL

Tesis presentada a la Dirección de Investigación y Postgrado
como parte de los requisitos para optar al grado de
Magister en Ciencias de la Ingeniería

Santiago de Chile, Enero 2010

AGRADECIMIENTOS

Agradezco todo el apoyo recibido durante el desarrollo de este trabajo. A mi profesro supervisor, Pedro Gazmuri, por sus consejos y guía tanto para la elaboración de esta tesis como para mi desarrollo personal y profesional.

Asimismo agradezco a los miembros de la comisión profesores José Pedro Prina, Rafael Riddell y José Manuel Carvajal, por sus numerosos aportes y comentarios para la mejora del tabajo.

Un especial agradecimiento a mi familia porque sé que sin su compañía se habría hecho muy difícil alcanzar esta meta y por darme la oportunidad de vivir esta experiencia y desarrollarme profesionalmente.

Finalmente agradezco a Dios por todo lo que día a día me regala y por la fortaleza entregada durante todo este tiempo.

INDICE GENERAL

AGRADECIMIENTOS iii

LISTA DE FIGURAS vii

LISTA DE TABLAS viii

RESUMEN ix

ABSTRACT x

1. INTRODUCCIÓN 1

1.1. Optimización Vía Simulación 1

1.1.1. Escenarios 4

1.2. Problema a Resolver 5

1.3. Estructura de la Tesis 8

2. MÉTODOS DE OPTIMIZACIÓN VÍA SIMULACIÓN 10

2.1. Métodos Basados en el Gradiente 10

2.1.1. Diferencias Finitas 11

2.1.2. Análisis de Perturbaciones - *Perturbation Analysis* 12

2.1.3. Razón de Verosimilitud - *Likelihood ratio* 12

2.1.4. Dominio de Frecuencia - *Frequency-domain* 13

2.2. Métodos de Trayectoria Muestreada - *Random Sampling* 14

2.3. Métodos de Superficie de Respuesta - *Response Surface Methodology* . . . 15

2.4. Métodos Heurísticos 16

2.4.1. Búsqueda Tabú - *Tabú Search* 17

2.4.2. Recocido Simulado - *Simulated Annealing* 17

2.4.3. Simplex Search 17

2.4.4. Algoritmos Evolutivos - *Evolutionary Algorithms* 18

3. ALGORITMOS EVOLUTIVOS 19

3.1. Codificación del Genoma 21
3.2. Población Inicial 22
3.3. Generación de Crías 23
3.3.1. Operadores de Selección 23
3.3.2. Operadores de Cruza 25
3.3.3. Operadores de Mutación 27
3.4. Función de Adaptación 28
3.5. Criterio de Término 29
3.6. Ranking y Selección 29
3.6.1. Métodos de dos etapas 31
3.6.2. Selección y Eliminación 34
3.6.3. Métodos completamente secuenciales 36

4. ALGORITMOS UTILIZADOS 41
4.1. nHGA 41
4.1.1. Población Inicial 42
4.1.2. Evaluación de adaptación 43
4.1.3. Generación de Crías 47
4.1.4. Etapa de Diversificación 48
4.1.5. Etapa de Intensificación 49
4.2. ISS 49
4.2.1. Población Inicial 51
4.2.2. Generación de Crías 51
4.2.3. Evaluación de Adaptación 53
4.2.4. Población Sobreviviente 55
4.2.5. Criterio de Parada 56
4.3. OptQuest 56

5. SISTEMAS DE SIMULACIÓN 59
5.1. Sistema 1: Abastecimiento de Combustible Ejército Turco 59

5.1.1. Descripción del Sistema 59
5.1.2. Problema de Optimización 62
5.2. Sistema 2: Mantención de Aviones de la Flota Finlandesa 63
5.2.1. Descripción del Sistema 63
5.2.2. Problema de Optimización 65
6. DESARROLLO DE EXPERIMENTOS 67
7. RESULTADOS 70
7.1. Sistema 1: Abastecimiento de Combustible Ejército Turco 70
7.2. Sistema 2: Mantención de Aviones de la Flota Finlandesa 76
8. CONCLUSIONES 80
BIBLIOGRAFIA 85
ANEXO A. PROBLEMA DE ABASTECIMIENTO DE COMBUSTIBLE 92

LISTA DE FIGURAS

1.1 Cantidad de estudios publicados sobre casos empíricos entre 1970-2000 5

3.1 Región de continuación para método SSM 40

5.1 Configuración del sistema de tanques de combustible. 60

5.2 Resultados con los niveles mínimos iguales para todos los tanques 62

6.1 Validación de programación algoritmo ISS en Matlab. 68

7.1 Desempeño algoritmo ISS en problema de abastecimiento de combustible . . . 71

7.2 evolución del mejor resultado obtenido de acuerdo al número de iteraciones . . 72

7.3 Medidas de desempeño para ISS en problema de flota aérea 77

7.4 Valor función objetivo conn OptQuest 78

LISTA DE TABLAS

5.1 Programa de mantenimiento según horas de vuelo 64

5.2 Tiempos requeridos en cada actividad de mantenimiento 65

5.3 Valor función de utilidad para configuraciones extremas 66

7.1 Medidas de desempeño para cada algoritmos utilizado 74

7.2 Mejor resultado obtenido con cada metodología 75

7.3 Resultado promedio obtenido con cada metología para problema de flota aérea finlandesa. 78

7.4 Mejor resultado con cada metología para problema de flota aérea finlandesa. . . 79

A.1Capacidad Tanques (m^3) 92

A.2Parámetros de la demanda en cada estación 93

RESUMEN

Frente al aumento en el uso de la simulación computacional han surgido numerosas metodologías de optimización vía simulación, a través de las cuales se busca encontrar los parámetros óptimos para cada sistema con el menor esfuerzo posible.

Gran interés han acaparado las metodologías heurísticas, especialmente los algoritmos evolutivos, que iteran imitando el comportamiento de las especies, hasta encontrar la mejor solución posible. Frente a este desarrollo surge la pregunta de cómo se desempeñan estas metodologías en problemas que pudieran representar el interés real de los modeladores.

El objetivo del presente trabajo es estudiar el comportamiento de algoritmos evolutivos seleccionados y comparar su comportamiento con el software de optimización OptQuest, para identificar las fortalezas y debilidades de cada metodología y los esfuerzos necesarios para aplicarlos en problemas reales.

El aumento en la dificultad de los problemas enfrentados se ve reflejado en la cantidad de variables de decisión, pasando de 3 variables en el caso de las comparaciones realizadas anteriormente entre estos algoritmos, a problemas con 20 y 7 variables respectivamente.

El desarrollo del trabajo muestra las dificultades, especialmente en cuanto a recursos computacionales, que se enfrentan para resolver problemas de mayor tamaño utilizando metaheurísticas para la resolución de problemas reales de optimización via simulación.

Se demuestra, en cuanto a los resultados respecto al valor de la función objetivo, que el algortimo nHGA obtiene mejores resultados que la estrategia ISS, sin embargo el primero requiere de más tiempo y mayor cantidad de evaluaciones de la función objetivo.

Al comparar los resultados obtenidos con el software de optimización OptQuest para Arena se observa que este último muestra un mejor desempeño, sin embargo, no es capaz de resolver cualquier problema debido a limitaciones en su funcionamiento.

Palabras Claves: Optimización, simulación, heurística.

ABSTRACT

Thanks to the increasing use of simulation, lots of optimization tools for simulation have been proposed which attempt to find the optimal parameters with the lest computational effort possible.

Heuristic methodologist had gain great attention, especially evolutionary algorithms. They mimic species behavior to find the best feasible solution. With this advances come the question about the performance of these techniques in problems that could represent a user interest.

The objective of this work is to study the behavior of some selected evolutionary algorithms and to compare them with the optimization software OptQuest and so, to identify each other strengths and weaknesses, and also to show the necessary effort to apply them on real problems.

The size rise in the problems is shown by the amount of decision variables in them, going form 3 in the originals parallels, to problems with 20 and 7 variables each.

This work development exemplifies the difficulties faced when methaheuristics are used to solve bigger problems, especially because of the great amount of computational resources needed.

Based on the results on the objective function, it is shown that the genetic algorithm nHGA gets better results than the evolutionary strategy ISS, however el first one needs more time than the second one.

OptQuest present better results than the other two techniques, but it is incapable of solving every type of problems due to limitations in its operation.

Keywords: Optimization, simulation, heuristic.

1. INTRODUCCIÓN

Simulación consiste en replicar el comportamiento de sistemas reales computacionalmente. Para su aplicación se utiliza un modelo computacional, a través del cual se miden diferentes medidas de desempeño claramente definidas, por medio de la observación del desarrollo o historia del comportamiento del modelo.

Su uso puede favorecer la toma de decisiones ya que permite, por ejemplo, observar el desempeño de diferentes sistemas considerando la incertidumbre inherente a los mismos, estudiar cómo afectarían cambios en la configuración del sistema en su comportamiento, estimar el funcionamiento de un sistema previo a su implementación o evaluar diseños alternativos con un bajo costo.

La simulación computacional es una de las metodologías más ampliamente utilizadas en gestión de operaciones. Pannirselvam, Ferguson, Ash y Siferd (1999) realizan una revisión de publicaciones en siete revistas entre los años 1992 y 1997, identificando la simulación como la segunda metodología más utilizada, superada exclusivamente por la optimización. Este interés se explica debido a que es una herramienta que permite estudiar sistemas donde no es posible obtener una expresión analítica para su desempeño, ya sea debido a su complejidad como a la existencia de incertidumbre.

Law y Kelton (2000) enumeran algunos de los principales usos y aplicaciones de la simulación, entre los que destaca el diseño y análisis de sistemas productivos, evaluación de armamentos militares, determinación de requerimientos para redes de comunicaciones, diseño y operación de sistemas de transporte, reingeniería de procesos de negocios, entre otros.

1.1. Optimización Vía Simulación

En los últimos años se ha producido un acercamiento entre la optimización y la simulación. A pesar de que hace varios años es tema de estudio, su uso en la práctica se ha ido

masificando. Fu (2002) presenta argumentos que demuestran esta realidad, entre los que se puede contar:

- La presencia actualmente de herramientas de optimización en prácticamente todos los softwares de simulación. Esto en constraste con la inexistencia de las mismas en 1990.
- La inclusión de capítulos completos sobre el tema en textos de simulación de uso masivo como Law y Kelton (2000).

Este acercamiento ha sido posible gracias al avance en la capacidad de procesamiento de los computadores, permitiendo realizar numerosas corridas de grandes sistemas en corto tiempo. Lo anterior, unido al desarrollo de metaheurísticas y mejoras en técnicas estadísticas, han hecho que las técnicas de optimización sean más aceptadas entre los académicos de la simulación (April, Better, Glover, Kelly & Laguna, 2006).

Un problema de optimización vía simulación es aquél en el cual se busca encontrar el set de parámetros que maximiza (minimiza) cierta función objetivo, cuyo valor sólo puede ser estimado a través de simulación.

Por lo tanto, suponiendo que se cuenta con una función objetivo $J(\theta)$ cuyo valor se desea optimizar, se plantea el problema de la siguiente manera:

$$\max_{\theta \in \Theta} \widehat{J}(\theta) \tag{1.1}$$

donde $\widehat{J}(\theta) = E\left[J(\theta)\right]$, θ representa el vector de variables de entrada y $\Theta \in \Re^d$ el espacio solución del problema, el cual, en el caso de interés, se asume continuo.

Azadivar (1999) resume las principales dificultades que enfrenta la optimización vía simulación versus un problema de programación no lineal general:

- No existe una expresión analítica para la función objetivo. Esto elimina la posibilidad de calcular gradientes locales.
- La función objetivo es una función estocástica con variables de decisión determinísticas. Se presenta, por lo tanto, un problema en la estimación de derivadas

locales aunque se busque solamente una aproximación. Es más, esto incluso perjudica el uso de enumeración completa, ya que, en base a una sola observación en cada punto, el mejor de éstos no puede ser determinado.

- Los programas de simulación requieren mayor esfuerzo computacional que evaluar funciones analíticas. Por lo tanto, la eficiencia de los algoritmos de simulación es crucial.
- Comunicar softwares de simulación con rutinas de optimización puede ser una tarea difícil.

Dentro de la optimización vía simulación existen dos procesos claves para el uso de recursos computacionales, la búsqueda y la evaluación. La primera consiste en el nivel de cobertura del espacio solución, la segunda es la evaluación de cada punto dentro del espacio de solución seleccionado en la búsqueda. Mientras más evaluaciones se tenga en cada punto más cercana será la estimación al valor real, sin embargo será necesario un mayor esfuerzo para las evaluaciones. Por lo anterior, será fundamental definir la utilización de los recursos de la mejor manera entre estos dos procesos.

Comúnmente el estudio de técnicas de optimización vía simulación se ha separado según las características del espacio de solución, ya sea si las variables de decisión son continuas, discretas o mixtas, así como si existe un número grande o pequeño de posibles soluciones. El estudio presentado a continuación se centrará en el área de optimización continua.

Las diferentes metodologías de optimización presentes en la literatura se pueden clasificar en los siguientes grupos, que posteriomente serán revisados:

- Métodos basados en el gradiente.
- Métodos de superficie de respuesta.
- Métodos de trayectoria muestreada.
- Métodos heurísticos.

Existe amplia literatura respecto al tema, pudiendo encontrarse revisiones de los avances en las diferentes metodologías (Azadivar, 1999; Fu, April & Glover, 2005; Carson &

Maria, 1997). Algunas de estas metodologías, como se verá más adelante, tienen la desventaja de ser dependientes del sistema a optimizar o de requerir conocimientos matemáticos específicos para su aplicación, lo que hace difícil su uso general.

1.1.1. Escenarios

Como ya se mencionó, existen diferentes metodologías de optimización vía simulación, sin embargo, debido a la variabilidad propia de los problemas, muchas pueden asegurar convergencia sólo después de muchas iteraciones, mientras otras sólo pueden asegurar convergencia a un óptimo local. Por esto la literatura prueba la convergencia utilizando diferentes escenarios de prueba.

Tal como lo evidencian Kabirian y Olafsson (2007) existen dos tipos de escenarios de prueba para los algoritmos de optimización. El primero consiste en efectivamente simular sistemas artificiales o reales para buscar el valor de las variables de decisión que optimicen cierta medida de desempeño. El segundo tipo utiliza funciones determinísticas a las cuales se les agrega un ruido aleatorio. Dichas funciones pueden variar en su complejidad, por ejemplo, a través de la presencia de múltiples óptimos locales. Las ventajas de cada tipo que menciona el trabajo son: el primer tipo se acerca más a la realidad, sin embargo, el experimentador no tiene conocimiento sobre la superficie de respuesta utilizada. El segundo tipo, en cambio, permite controlar una gran variedad de superficies de respuesta.

Pasupathy y Henderson (2006) reconocen la necesidad de contar con un set estándar de escenarios de prueba para comparar el desempeño de los algoritmos y dan ciertas guías para alcanzarlo.

A pesar de que ambos tipos de escenarios han sido utilizados en la literatura, es común encontrarse con pruebas del segundo tipo. Para el primero, en cambio, usualmente se encuentran escenarios de prueba con no más de tres variables de decisión, como por ejemplo sistemas del tipo M/M/1 o similares. Se cree, sin embargo, que algunos de los algoritmos que muestran rápida convergencia con estos métodos podrían ver seriamente afectado su desempeño para escenarios de prueba que utilicen simulación de sistemas complejos.

1.2. Problema a Resolver

El objetivo del presente trabajo es estudiar el desempeño de dos algoritmos de optimización cuando se quieren resolver problemas de mayor magnitud a los utilizados como escenarios de prueba, para así definir si efectivamente mantienen sus diferencias comparativas y fortalezas individuales al enfrentarse a un sistema de simulación que pudiera representar un problema de aplicación real.

En los últimos años ha aumentado significativamente la literatura sobre el uso de simulación en casos prácticos. Shafer y Smunt (2004) presentan un estudio sobre estos casos en 20 revistas del área, destacando el aumento progresivo en los años, tal como lo ejemplifica la figura 1.1.

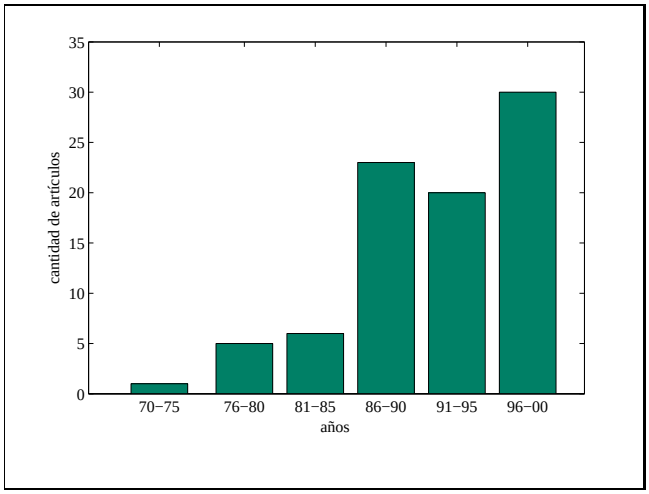

FIGURA 1.1. Cantidad de estudios publicados sobre casos empíricos entre 1970-2000

A pesar de lo anterior, las dificultades que plantea el uso de algunas metodologías de optimización, creó una brecha entre el uso de simulación como herramienta para la investigación operacional y la inclusión de optimización en estos mismos. Entre otras razones porque gran parte de las metodologías de optimización requieren de un conocimiento matemático del sistema a optimizar difícil de alcanzar para un usuario promedio.

Sin embargo desarrollos recientes han cambiado este panorama. El avance en el campo de metaheurísticas, que mejora métodos matemáticos tradicionales con el uso de inteligencia artificial y métodos que imitan procesos biológicos, han llevado a la creación de estructuras de optimización que pueden guiar exitosamente evaluaciones complejas para encontrar el valor óptimo de las variables de decisión, evitando quedar atrapados en óptimos locales (Laguna & Marti, 2002). Y que tienen la virtud de ser métodos de aplicación general que utilizan el sistema de simulación como una caja negra.

Estas técnicas son las más utilizadas actualmente en los softwares comerciales que incluyen aplicaciones de optimización, lo que marca una diferencia respecto al énfasis en el mundo académico, el cual se centra especialmente en metodologías específicas como por ejemplo aquellas basadas en el gradiente, debido especialmente a que en ellas se cumplen ciertas condiciones que aseguran convergencia a un óptimo, ya sea global o local.

Tal como lo plantea Fu (2002), debe existir una transición desde la optimización que asegura convergencia, aunque en un contexto de difícil aplicación, hacia soluciónes más manejables aplicables a problemas prácticos. Esta afirmación refleja las diferencias entre la investigación y la práctica, donde se enfrentan aproximaciones que proveen soluciones rápidas y robustas sin garantizar desempeño (basadas en heurísticas, sin análisis estadístico), con aproximaciones matemáticas rigurosas (que aseguran convergencia o cierta probabilidad de selección correcta) que dominan la literatura académica.

Existe variada literatura sobre el uso de metodologías de optimización en diferentes problemas de simulación (Kleijnen & Wan, 2007; Lee, Khoo & Yin, 2000; Dummler, 1999), algunos de ellos buscan probar la convergencia de métodos basados en el gradiente, mientras que los restantes utilizan, en su mayoría, herramientas de optimización disponibles comercialmente con los softwares de simulación. Sin embargo existe muy poca información sobre qué heurística presenta un mejor desempeño en el momento de optimizar un problema real.

Lo anterior, combinado con el aumento en la aceptación de las técnicas de optimización vía simulación, demuestra la necesidad de conocer el desempeño real de los algoritmos de

optimización bajo condiciones de simulación que pudieran, efectivamente, entregar información relevante a los tomadores de decisiones.

En los últimos años se ha hecho patente la necesidad de contar con mayor información sobre el desempeño de este tipo de algoritmos en casos reales. Así se muestra a través de las palabras de John Carson (Fu et al., 2000) al indicar que en su opinión lo que se necesita actualmente es que un investigador pruebe los métodos bajo variados modelos de simulación.

Desde hace algunos años se ha plantedo la creación de una serie de ensayos para probar los diferentes mecanismos de optimización. Estos ensayos deben, idealmente, incluir problemas que sean representativos de aplicaciones reales. Sin embargo, dados los avances en los algoritmos, es probable que se expanda dicho rango de problemas de interés práctico. Por lo tanto, también se deben incluir problemas que fuercen los límites de las metodologías que se están probando (Whitley, Rana, Dzubera & Mathias, 1996).

Las razones anteriores son las que explican y reflejan la necesidad de contar con mayor información sobre cómo se desenvuelven ciertos algoritmos de optimización con problemas con un mayor número de variables de decisión.

Adicionalmente, mediante el desarrollo de estas pruebas, se espera reconocer y comparar las limitaciones y características específicas de cada metodología; determinar las condiciones bajo las cuales será recomendable utilizar cada una de las heurísticas; entregar información relevante sobre el uso de optimización vía simulación en experimentos de mayor tamaño; y, dentro de las posibilidades, proponer y probar posibles modificaciones a las heurísticas para mejorar su desempeño.

Para esto se utilizarán metaheurísticas desarrolladas en los últimos 5 años y, adicionalmente, sus resultados se compararán con el desempeño de uno de los softwares de simulación más utilizado comercialmente en la actualidad, OptQuest[1] [2].

[1]Tal como se afirma en (Rogers, 2002).
[2]Software desarrollado por Optimization Technologies, Inc.

Entre las metaheurísticas propuestas se encuentra el trabajo desarrollado por Olguin (2008), que consiste en un algoritmo genético en dos etapas con diferenciación entre padres masculinos y femeninos. En dicho trabajo se muestra la mejoría alcanzada con esta heurística respecto a la desarrollada por Buchhloz y Thümmler (2005), consistente en una estrategia evolutiva con población elite.

Los experimentos realizados para comparar estas dos heurísticas son similares a lo que predomina en la literatura de este tipo de metodologías. En primer lugar utiliza una función esfera con ruido, de la cual se conoce su único óptimo global, y en segundo lugar una línea de producción conformada por tres servidores en serie con bloqueo, para la cual se busca optimizar una función de utilidad dependiente del *throughput*[3] del sistema.

Por lo tanto, debido a que ya se cuenta con información que compara ambos mecanismos a través de problemas sencillos, a lo reciente de sus propuestas, a que ambas son heurísticas de aplicación general para problemas de optimización continuas y a que representan dos variantes en creciente desarrollo de los algoritmos evolutivos, estas serán las dos metodologías a utilizar en el trabajo.

En razón de lo anterior se postulan dos ejemplos de la literatura que utilizan simulación en casos reales, para los cuales se buscará optimizar cierta medida de desempeño.

1.3. Estructura de la Tesis

Una vez establecido lo anterior, el resto de la tesis se organiza de la siguiente manera: El capítulo 2 cubre las principales metodologías utilizadas en optimización via simulación, los cuales pueden agruparse según métodos basados en el gradiente, métodos de trayectoria muestreada, métodos de superficie de respuesta y métodos heurístico.

El capítulo 3 se centra en las estrategias evolutivas, un caso particular de métodos heurísticos que buscan imitar el comportamiento evolutivo de las especies, presenta el marco teórico general de estas estrategias, diferenciando entre algoritmos genéticos y algoritmos evolutivos.

[3]Cantidad de productos por unidad de tiempo.

A continuación, en el capítulo 4, se detallan los algoritmos utilizados en el trabajo, particularmente las metodologías ISS y nHGA, así como una revisión del funcionamiento del software de optimización OptQuest para Arena.

El capítulo 5 describe los sistemas a optimizar, así como las funciones objetivos y restricciones de cada uno de ellos. Mientras que en el capítulo 6 se describe brevemente el trabajo realizado para el desarrollo de los experimentos.

El capítulo 7 presentan y comparan los resultados de las pruebas realizadas con cada uno de los algoritmos para cada uno de los problemas seleccionados.

Finalmente en el capítulo 8 se entregan las conclusiones del presente trabajo, así como las posibles investigaciones futuras en el área y los alcances de esta investigación.

2. MÉTODOS DE OPTIMIZACIÓN VÍA SIMULACIÓN

El objetivo de la optimización vía simulación es encontrar la combinación de parámetros de entrada que optimicen una función de las medidas de desempeño del sistema simulado.

En los sistemas de simulación se distingue entre variables controlables y no controlables. El énfasis en optimización está puesto en las primeras, ya que son parte esencial del diseño de los sistemas, sin embargo, puede ser de interés analizar también las segundas.

Tal como se menciona en Law y Kelton (2000), el desafío en la optimización vía simulación es decidir qué configuraciones alternativas simular y cómo evaluar y comparar los resultados.

A continuación se presenta una breve revisión de algunos de los métodos utilizados para la optimización de parámetros continuos.

2.1. Métodos Basados en el Gradiente

Buscan imitar algoritmos de descenso ya conocidos de programación matemática, a través de estimaciones del gradiente en cada punto de evaluación. La dificultad en su aplicación radica en que, como la función objetivo depende de medidas de desempeño de la simulación, no se contará con una expresión analítica para el gradiente en cada punto de evaluación.

Para problemas estocásticos los procedimiento más utilizado son de la clase de aproximación estocástica (*Stochastic Approximation, SA*), los cuales entregan una estimación del gradiente de la función objetivo, cuya expresión analítica no puede ser conocida.

Los algoritmos de aproximación estocástica tienen su origen en el llamado algoritmo Robbins-Monro (Robbins & Monro, 1951). En ellos se realiza, en cada etapa, un movimiento dentro del espacio solución en la dirección del gradiente de la función de respuesta. Por lo tanto, se utiliza una fórmula recursiva, según la cual la configuración actual del sistema se

actualiza utilizando una aproximación del gradiente según la forma:

$$\theta_{n+1} = \theta_n + a_n \nabla(f(\theta_n)) \tag{2.1}$$

donde los a_i representan una secuencia de números reales positivos que deben cumplir las siguientes condiciones:

$$\sum_{n=1}^{\infty} a_n = \infty, \quad \sum_{n=1}^{\infty} a_n^2 < \infty$$

La condición anterior en conjunto con que el sesgo de la estimación del gradiente se vaya a cero son condiciones suficientes para que el método converja a un óptimo local. Generalmente se utiliza la secuencia $a_n = a/n$.

Para su aplicación será necesario contar con el gradiente ($\nabla f(\theta)$) en cada posible solución. Sin embargo, frente a la ausencia de una función analítica, su valor deberá ser estimado y dicha estimación estará sujeta a la variabilidad de la salida de la simulación.

El desempeño de esta metodología estará fuertemente ligado a la elección de la serie a_n y a la calidad de la estimación del gradiente, donde también jugará un papel fundamental la cantidad de evaluaciones del modelo necesarias para su estimación. Una completa revisión puede ser encontrada en Kushner y Yin (1997).

A continuación se presentan algunos de los métodos más utilizados para la estimación del gradiente en base a la revisión realizada por Andradottir (1998a).

2.1.1. Diferencias Finitas

Consiste en estimar cada una de las derivadas parciales de f en cada punto θ_i. Suponiendo $g(\theta) = \nabla f(\theta)$ se tendrá para cada coordenada i de $g(\theta)$

$$\widehat{g}_i(\theta) = \frac{\widehat{f}(\theta + ce_i) - \widehat{f}(\theta)}{c}$$

con e_i el vector unitario en la dimensión i y c un valor pequeño.

Este mecanismo necesitará, si se cuenta con d variables de decisión, $d+1$ estimaciones de $\widehat{f}$.

Para reducir el sesgo del estimador anterior, se pueden utilizar diferencias centrales donde:

$$\widehat{g}_i(\theta) = \frac{\widehat{f}(\theta + ce_i) - \widehat{f}(\theta - ce_i)}{2c}$$

en este caso se requerirán $2d$ estimaciones, exigiéndose un mayor esfuerzo computacional.

Es importante notar que, para estimar el gradiente de manera más exacta, se debe utilizar c lo más pequeño posible, sin embargo, su varianza será proporcional a $1/c^2$, por lo tanto, existirá una contraposición entre mejorar la estimación y reducir su varianza. Por lo anterior, se propone utilizar números aleatorios comunes para reducir la variabilidad de la estimación. La principal ventaja de este método es que su aplicación no depende de la estructura del sistema.

2.1.2. Análisis de Perturbaciones - *Perturbation Analysis*

Esta clase de métodos permiten estimar el gradiente a partir de una sola réplica. Entre ellos destaca principalmente el tipo análisis de perturbación infinitesimal, el cual se fundamenta en que frente a un cambio marginal en uno de los parámetros de entrada, la lógica de los eventos no se modifica. Es decir, luego de una perturbación infinitesimal de uno de los parámetros de entrada, la secuencia de eventos será la misma que en la trayectoria original. Por lo tanto, la sensibilidad de la respuesta del sistema hacia ese parámetro puede ser estimada trazando su trayectoria de propagación a través del sistema.

Existen numerosas investigaciones sobre este tema, pudiendo encontrarse revisiones completas por ejemplo en Glasserman (1991). La principal desventaja es que es fuertemente dependiente del sistema a modelar. Suri (1989) y Fu y Hu (1996) revisa los avances en el área, presenta ejemplos de aplicación y casos en los cuales no es posible utilizar esta metodología.

2.1.3. Razón de Verosimilitud - *Likelihood ratio*

Consiste en expresar el gradiente $g(\theta)$ como una esperanza que puede ser estimada a través de la simulación y que es función de los parámetros del sistema o de los parámetros de la simulación. Glynn (1990) presenta un análisis de esta metodología.

Supongamos que se toma como parámetro de la simulación la variable aleatoria de input X, entonces asumiendo por simplicidad:

$$f(\theta) = E_\theta[h(\theta, X)]$$

suponiendo que la variable aleatoria $X \tilde{} F_\theta$ tiene una función de densidad f_θ entonces se tendría:

$$f(\theta) = \int h(\theta, x) f_\theta(x) dx$$

Andradottir (1998b) muestra que, desarrollando la expresión anterior, se puede estimar el gradiente a partir de la simulación como:

$$g(\theta) = f'(\theta) = E[h'(\theta, X) + h(\theta, X)\frac{f'_\theta(X)}{f_\theta(X)}] \tag{2.2}$$

por lo tanto, se puede estimar $g(\theta)$ generando $X_1, ..., X_N$ observaciones independientes con distribución F_θ y a partir de la simulación calcular:

$$\widehat{g}(\theta) = \sum_{i=1}^{N} [h'(\theta, X_i) + h(\theta, X_i) f'_\theta(X_i)/f_\theta(X_i)]/N$$

Para el caso de estos dos últimos estimadores del gradiente es importante destacar que será necesario tener información sobre valores internos (o variables aleatorias de entrada) de la simulación, los cuales deben ser recolectados por el modelador.

Adicionalmente, ambos métodos son altamente dependientes del problema y exigen un conocimiento matemático superior del modelador para poder generar los estimadores necesarios a partir de cada problema en particular. Sin embargo, el método de razones de verosimilitud ha podido ser aplicado en mayor variedad de problemas que el análisis de perturbaciones no puede resolver.

2.1.4. Dominio de Frecuencia - *Frequency-domain*

Este método consiste en hacer oscilar el valor de las variables de decisión de manera sinusoidal durante una única corrida del sistema. A partir de esto se puede estimar la

sensibilidad de la medida de desempeño a cada uno de estos valores, lo que implica estimar también el gradiente.

Por ejemplo, como lo plantea Andradottir (1998b), si se desea estimar $g(\theta) = \nabla f(\theta)$, con $\theta = (\theta_1, ..., \theta_d)$ se tendrá que el valor de cada parámetro de entrada en un instante t del reloj de la simulación será $\theta_i(t) = \theta_i + \alpha_i \sin(\omega_i t)$. Luego, la medida de desempeño puede ser aproximada, en la vecindad de θ, usando una función polinomial de respuesta a los parámetros de entrada. Dicha aproximación puede ser utilizada para derivar estimadores, en el dominio de la frecuencia, de $g(\theta)$.

La metodología de dominio de frecuencia fue postulada por primera vez como herramienta de inspección para simulación de eventos discretos con parámetros de entrada continuos en Shruben y Cogliano (1987). Posteriormente, su uso fue extendido para estimar el gradiente de la función de salida.

2.2. Métodos de Trayectoria Muestreada - *Random Sampling*

La presentación formal del método puede ser encontrada en Gurkan, Yonca y Robinson (1994). Busca aplicar técnicas de optimización determinística a problemas de simulación. Para esto el valor esperado de la función objetivo es estimado a través del promedio de una gran cantidad de observaciones en cada punto del espacio solución.

Suponiendo que $\widetilde{f}_j$ representa el valor estimado de f en la j-ésima réplica del sistema, la media muestral sobre n replicas será:

$$\widehat{f}_n(\theta) = \frac{1}{n} \sum_{j=1}^{n} \widetilde{f}_j(\theta)$$

Si cada uno de los $\widetilde{f}_j$ son estimadores insesgados i.i.d. de f y n es un valor suficientemente grande, entonces, por la ley de los grandes números se tendrá:

$$\widehat{f}_n(\theta) \to f(\theta) \quad \text{con probabilidad 1.}$$

Luego, para un valor suficientemente grande de n, se busca optimizar la función determinística $\widehat{f}_n$. Para esto se pueden utilizar métodos de descenso, estimando el gradiente con diferentes métodos de los ya vistos. Robinson (1996) presenta una justificación matemática de su funcionamiento y entrega condiciones para su convergencia.

La principal ventaja de este método es que permite resolver problemas con restricciones complejas utilizando técnicas de optimización determinística.

2.3. Métodos de Superficie de Respuesta - *Response Surface Methodology*

A través de varias estimaciones de la respuesta y de una serie de modelos de regresión, se busca obtener una aproximación a una relación funcional entre las variables de entrada y la función objetivo del problema de optimización.

Esta metodología fue propuesta por Box y Wilson (1951) y desde entonces se han realizado numerosos estudios sobre su aplicación (Hood & Welch, 1993). En términos generales consiste en lo siguiente (Kleijnen, 2008):

- Comienza con la selección de un punto de partida y luego explora una vecindad de dicho punto a través de simulación.
- Luego se aproxima una superficie de respuesta a los valores obtenidos, en función a las variables de decisión, a través de una aproximación polinomial de primer orden.
- A partir de la aproximación anterior se realiza una estimación del gradiente y se realiza un movimiento en el espacio solución en la dirección del paso ascendente del gradiente.
- Es necesario dar una magnitud al paso en la dirección indicada. Una primera aproximación para esto es que el modelador lo decida intuitivamente. Si producto del tamaño del paso se obtiene una solución de menor calidad, se deberá probar con un paso menor.
- Después de una determinada cantidad de pasos en la dirección del gradiente, será necesario suplir el deterioro de la estimación de primer orden.

- Cuando se cree haber alcanzado la región del máximo, se realiza en ella una regresión de segundo orden. Para, a partir de ésta, estimar el óptimo a través de diferenciación o de análisis canónico.
- Si alcanza el tiempo, para evitar caer en mínimos locales, se recomienda reiniciar la búsqueda a partir de un nuevo punto de partida.

Cabe mencionar que el método es fuertemente dependiente del punto de partida escogido y que puede caer en óptimos locales, sin barrer todo el posible espacio solución.

Existen otras variaciones al método anterior, como por ejemplo, realizar una estimación de la superficie de respuesta para todo el espacio de solución, lo que es conocido como metamodelo, para luego aplicar métodos de programación matemática determinística a dicha aproximación.

Adicionalmente Kleijnen (2008) propone variaciones que le permiten tratar problemas con más de una respuesta del modelo de simulación, donde una de ellas es tratada como la función a optimizar y las demás pueden formar parte de restricciones del problema. Dicho método es conocido como *generalized response surface methodology*.

Carson y Maria (1997) sostienen que, en general, este método requiere menos evaluaciones del sistema en comparación a muchos métodos basados en el gradiente. El principal inconveniente parece ser la cantidad excesiva de puntos de simulación en un área antes de pasar a otras regiones del espacio de búsqueda. Este problema puede ser aún mayor si la cantidad de variables de entrada es grande. Además no ha sido implementado en ninguno de los softwares comerciales de optimización vía simulación.

2.4. Métodos Heurísticos

En los últimos años ha crecido el interés por desarrollar métodos heurísticos que utilicen la simulación como una caja negra para evaluar el valor de la función objetivo. Por lo anterior, serán de aplicación general para cualquier sistema sin importar sus características.

La capacidad de aplicación general descrita anteriormente presenta la desventaja de que el método no aprovecha de manera eficiente la estructura del sistema, lo cual podría

mejorar su velocidad de resolución. La mayoría de estas técnicas buscan un balance entre la exploración del espacio solución y la evaluación de los resultados.

A continuación se presenta una breve descripción de algunos de estos métodos, basada principalmente en el trabajo de Carson y Maria (1997) y Fu et al. (2005).

2.4.1. Búsqueda Tabú - *Tabú Search*

Procedimiento desarrollado por Fred Golver que busca evitar quedar atrapado en óptimos locales. Para esto, se mantiene un listado, con una cantidad fija de elementos, que representa movimientos *tabu*, es decir, movimientos prohibidos para el próximo punto a evaluar en el espacio solución. Adicionalmente, puede ser combinado con otros métodos de optimización para suplir la dificultad que producen los óptimos locales.

2.4.2. Recocido Simulado - *Simulated Annealing*

Es un método de búsqueda estocástica que busca imitar el proceso de enfriamiento de los cristales, en el cual una aleación es enfriada gradualmente para alcanzar un estado de mínima energía.

Consiste en un método de descenso que, para evitar quedar atrapado en óptimos locales, puede aceptar movimientos en direcciones que no sean las de máximo descenso. Para lo anterior, se define una probabilidad de aceptar estos movimientos, que busca imitar el comportamiento de la temperatura en un proceso de enfriamiento y que, por lo tanto, va disminuyendo a medida que la heurística avanza. Una descripción completa puede encontrarse en Eglese (1990).

2.4.3. Simplex Search

Este método comienza con $p + 1$ vértices en un simplex en el espacio solución y continúa descartando el peor de estos puntos y agregando uno nuevo, determinado por la reflexión del peor punto a través del centroide de los vértices restantes. Fue propuesto por Nelder y Mead (1965).

La mayor dificultad para su aplicación es determinar el peor de los puntos, ya que, debido al ruido en la evaluaciones, se pueden eliminar puntos que sean mejores que otros. Para esto se comparan las respuestas en cada vértice con métodos estadísticos.

Una extensión es el método *Complex search*, que busca adaptarlo a problemas con restricciones.

2.4.4. Algoritmos Evolutivos - *Evolutionary Algorithms*

Corresponden a una clase de métodos de optimización continua inspirados en la evolución natural de las especies. Estos consisten en heurísticas de búsqueda basadas en una población que utilizan variación aleatoria y selección.

Generalmente, en cada nueva iteración, se generan λ crías a partir de un conjunto de μ padres, a través de diferentes técnicas de selección, cruza y mutación. Cada individuo tiene asociado un valor de adaptación *(fitness)*, que, usualmente, corresponde al valor de la función a optimizar, a partir de la cual se seleccionan los mejores individuos para ser los padres en la siguiente iteración.

Entre los diferentes tipos de algoritmos evolutivos se cuentan: los algoritmos genéticos, las estrategias de evolución y la programación evolutiva. En el capítulo siguiente se profundizará en estas metodologías poniendo especial énfasis en las dos primeras.

3. ALGORITMOS EVOLUTIVOS

Como ya se mecionó, los algoritmos evolutivos buscan imitar el proceso de evolución de las especies bajo la teoría de que los individuos más fuertes de una población tendrán mayores posibilidades de sobrevivir. Los individuos más fuertes serán aquellos mejor adaptados, donde la adaptación se define como una medida de la calidad de la solución que entrega cada individuo al problema, y tendrán mayores posibilidades de sobrevivir.

En forma general estos algoritmos se basan en una población de individuos, cada uno de los cuales representa un punto en el espacio solución del problema. Dicha población es inicializada arbitrariamente para luego evolucionar hacia áreas mejores dentro del espacio solución.

Para la evolución se cuenta con una población inicial de la cual se escogerá un número establecido de padres a través del proceso de selección. Éstos darán origen a crías a través de procesos aleatorios determinados por los mecanismos de mutación y cruza. A continuación, el sistema entregará información sobre la adaptación de cada uno de los individuos, en base a lo cual, aquellos mejor adaptados tendrán mayor posibilidad de sobrevivir y ser escogidos como padres en la etapa siguiente.

El ciclo anterior se repetirá hasta que se cumpla una determinada condición de término, entre las que se puede contar alcanzar cierta cantidad establecida de iteraciones, que la mejor solución no cambie durante un período determinado, etc.

Según lo explicado anteriormente se presenta un seudocódigo general de un algoritmo evolutivo simple:

Algoritmo 1: Algoritmo evolutivo

generación población inicial;

while *no se cumpla la condición de término* **do**

    Evaluación de los individuos;

    Definir población actual;

    Selección;

    Cruce;

    Mutación;

end

Entrega el mejor individuo de la población;

Existen diferentes tipos de algoritmos evolutivos, entre los que destacan los algoritmos genéticos, propuestos por Holland en la década del 60 (Holland, 1975), y las estrategias de evolución, desarrolladas por Rechenberg y Schwefel en la misma época en Alemania (Rechenberg, 1965). Un tercer tipo es la programación evolutiva o genética, que presenta soluciones a los problemas mediante una población de máquinas de estados finitos o mediante programas y la nueva población se crea a partir de la mutación aleatoria de los programas padre. Una descripción completa de esta metodología se puede encontrar en Bäck (1996).

De manera general para los algoritmos evolutivos, se utiliza una notación que indica la cantidad de padres (individuos que sobreviven de la población) y crías existentes en cada iteración y las diferentes estrategias que se pueden adoptar con dicha población. Así, la representación $(\mu + \lambda)$-ES indicará que se trata de una estrategia evolutiva con μ padres, los cuales generan λ crías y donde la población para la siguiente iteración corresponderá a los μ individuos mejor adaptados del conjunto padres+crías. En cambio, la representación (μ, λ)-ES seleccionará como sobrevivientes a los μ individuos mejor adaptados solamente entre las crías.

Existen numerosos artículos, tanto sobre estrategias evolutivas como sobre algortítmos genéticos, que pueden entregar una visión más profunda sobre su funcionamiento (Bäck, 1996; Beyer & Schwefel, 2002).

A continuación se presentan los principales componentes de los algoritmos evolutivos y se explica cómo se aplican generalmente para el caso de algoritmos genéticos y de las estrategias evolutivas.

3.1. Codificación del Genoma

Cada individuo de la población es representado a través de su codigo genético único o genoma, el cual contiene las características particulares de cada uno de ellos, como por ejemplo, valores factibles para cada una de las variables de decisión del problema.

Existen diferentes maneras de codificar esta información, ya sea a través de código binario, código gray o codificación real. Cada una con sus ventajas y desventajas.

Para el caso de estrategias evolutivas, el énfasis en la codificación del genoma está puesto en la codificación real. Además, para este tipo de metodologías, el código genético de un individuo de la población puede contener no sólo las coordenadas en el espacio de búsqueda, sino que también información sobre la mutación de dicho individuo, lo que puede estar dado, por ejemplo, por características de la varianza de su mutación o de la rotación de los puntos en el espacio solución al mutar (identificado con la covarianza entre los diferentes parámetros). Así, para estos casos, la optimización no sólo ocurre en base a las variables de decisión, sino que también en estos parámetros estratégicos de acuerdo a la topología de la función objetivo. Este hecho es conocido como la capacidad de auto adaptación. (Müller, Sbalzarini, Walther & Koumoutsakos, 2001)

En el caso de los algoritmos genéticos, tradicionalmente se ha utilizado codificación binaria. Esto simplifica el trabajo en caso que las variables puedan ser representadas por un uno o un cero. Sin embargo, para casos de espacio solución más elaborados, cada parámetro está asociado a una cadena de una cantidad fija de bits que representan algún valor en el espacio solución, por lo tanto, para la aplicación del algoritmo será necesario

contar con codificaciones y decodificaciones que pueden ser complicadas. Adicionalmente, para este tipo de codificación en el caso de variables continuas, la posible cobertura del espacio solución estará determinada por un número fijo de individuos (según la cantidad de bits utilizados para la representación), de esta manera, no se podrá cubrir todo el espacio solución y la solución óptima podría no estar abarcada en dicho grupo.

Por lo anterior, en los últimos años ha aumentado el uso de codificación real en los algorítmos genéticos, donde el genoma es modelado con un vector cuyas coordenadas son números reales que representan la solución factible asociada a cada individuo. Diversos trabajos han utilizado este enfoque demostrando la eficiencia de dicha codificación para problemas particulares (Bessaou & Siarry, 2001; Hwang & He, 2006; Tsutsui, Ghosh, Corne & Fujimoto, 1997).

3.2. Población Inicial

El tamaño de la población será un factor muy importante en la convergencia de los algoritmos evolutivos, así, si el número de individuos es mayor, se explorarán más zonas del espacio solución, sin embargo esto traerá asociado un costo computacional mayor.

Tal como se afirma en Kabirian y Olafsson (2007), el tiempo computacional en cada iteración de un algoritmo de optimización vía simulación se puede dividir en dos partes: el tiempo requerido para proponer nuevas posibles soluciones y el tiempo de evaluación de las corridas de cada una de estas posibles soluciones. Ellos sostienen que cuando se utiliza simulación para estimar la función objetivo, el tiempo de la primera parte puede ser mucho menor que el de la segunda ya que, ejecutar un modelo de simulación para un sistema real, generalmente consume mucho tiempo. Por lo anterior, se debe realizar un equilibrio entre la exploración del espacio solución y la explotación de cada una de las posibles soluciones.

Otra decisión relacionada con la población inicial es la creación de esta misma. Una vez definido el tamaño de la población, será necesario contar con una población inicial para el desarrollo de los algoritmos. Las características de estos individuos también tendrán influencia sobre la convergencia de cada algoritmo.

Se proponen diferentes metodologías para las creación de la población inicial, como por ejemplo, crearla a partir de buenas soluciones conocidas o generarla de manera aleatoria distribuidas uniformenente en el espacio solución. Si éste no es cubierto homogéneamente por la población inicial se correrá el riesgo de una convergencia prematura a un óptimo local, lo que dificultaría la identificación de óptimos globales.

Para lograr mayor homogeneidad en la cobertura del espacio solución se propone utilizar un concepto de vecindad en cada uno de los individuos creados, donde se define un área, llamada vecindad, alrededor de cada individuo de la población y se impone como restriccióno que ningún individuo puede pertenecer a la vecindad de otro miembro de la población. Hu (1992) propone, bajo este concepto, una vecindad esférica, es decir, se define la vecindad como la "esfera" centrada en el individuo s y con radio ϵ que contiene todos los individuos s' tales que $\|s' - s\| \leq \epsilon$. Chelouah y Siarry (2000) utiliza este concepto en un algoritmo genético.

3.3. Generación de Crías

Se conoce como operadores genéticos a aquellos encargados de generar las crías, o nuevas posibles soluciones, dentro del espacio solución.

3.3.1. Operadores de Selección

Los operadores de selección escogen a los individuos de la población actual que serán utilizados para la reproducción. Adicionalmente, en algunos casos, determina cuántas crías producirá cada individuo seleccionado.

Para esto, existen diferentes técnicas, la mayoría de las cuales buscan asegurar que los mejores individuos tengan mayor probabilidad de ser seleccionados. Se puede encontrar un análisis y comparación de estas mismas en (Goldberg & Deb, 1991).

Entre las metodologías más utilizadas se pueden nombrar las siguientes, donde las dos finales favorecen a los individuos mejor adaptados:

3.3.1.1. Aleatoria

Consiste en seleccionar aleatoriamente a los individuos para la reproducción. En este caso todos los miembros de la población actual tienen igual probabilidad de ser elegidos.

3.3.1.2. Método de Ruleta

Busca favorecer a aquellos individuos de la población mejor adaptados. Para esto se asigna a cada individuo una probabilidad de ser seleccionado que depende de su función de adaptación, mientras mejor sea dicho valor, mayor será la probabilidad.

Por lo tanto, para una población de n individuos, se define la probabilidad P_{si} de seleccionar al individuo i como:

$$P_{si} = \frac{f_i}{\sum_{j=1}^{n} f_i}$$

donde f_i corresponde al valor de la función de adaptación del individuo i.

A continuación se calcula la probabilidad acumulada de selección P_{ai} de cada individuo como:

$$P_{ai} = \sum_{j=1}^{i} P_{sj} \tag{3.1}$$

Finalmente, se genera un número aleatorio entre 0 y 1 y se selecciona como padre al individuo que tenga la probabilidad acumulada inmediatamente mayor a este número.

Una variación a este tipo de selección es la selección sigma, en la cual la probabilidad de seleccionar a un individuo dependerá, no sólo de su adaptación, sino que también del nivel de adaptación medio de la población y de la desviación estándar de la población completa.

3.3.1.3. Torneo

En este caso se elige aleatoriamente un número determinado de individuos, los cuales luego son puestos a competir entre sí, a través de la función de adaptación. Del grupo elegido inicialmente se seleccionará como padre a aquél que tenga el mayor valor en la función de adaptación.

Como complemento han surgido modificaciones a la definición original, por ejemplo, elegir un grupo inicial de dos individuos y ocupar una cierta probabilidad de que el seleccionado sea el mejor adaptado de los dos y luego, a través de un número aleatorio, decidir si se queda el mejor de los dos o aquel con peor adaptación.

3.3.2. Operadores de Cruza

Este operador busca combinar la información genética de dos o más padres para generar nuevos individuos.

Se pueden diferenciar dos grandes grupos de operadores de cruza, aquellos para codificación binaria y los que son para algoritmos con codificación real. Ambos tipos pueden incluir, antes de realizar la cruza, una probabilidad de que el nuevo individuo sea producto de una cruza o no. Lo anterior se traduce en la práctica en una decisión basada en un número aleatorio obtenido entre $[0, 1]$ donde, si este valor es menor a la probabilidad de cruza, se realizará la cruza y si es mayor no habrá cruza.

3.3.2.1. Cruza de individuos con codificación binaria

Para el caso de dos individuos con codificación binaria el operador más sencillo fue introducido por Holland y es conocido como cruza de un punto. En él se selecciona aleatoriamente un punto dentro de la cadena de bits que representa a cada individuo y se intercambian entre ambos padres todos los bits que están a la derecha de este punto. Esto da como resultado dos crías de las cuales se puede seleccionar una aleatoriamente.

Con el tiempo han surgido modificaciones a este operador, por ejemplo, su generalización en la cruza de múltiples puntos, donde un número fijo de posiciones en la cadena de bits son seleccionados para luego intercambiar los bits de los padres sección por medio de las que se forman entre estos puntos. Llevando el caso anterior al máximo posible se llega a la cruza uniforme, donde se decide bit a bit si se realiza el intercambio basado en una decisión aleatoria.

Existen más modificaciones al caso original y cada vez surgen nuevas propuestas para este operador. Númerosos estudios empíricos han demostrado que la cruza de un punto

parece ser inferior a otros operadores (Eshelman, Caruna & Schaffer, 1989; Schaffer, Caruna, Eshelman & Das, 1989)

3.3.2.2. Cruza de individuos con codificación real

Para el caso de codificación real también existen diferentes operadores, algunos de los cuales utilizan dos padres para la cruza y otros que eligen en primer lugar un padre en torno al cual se realizará la cruza y luego, para cada componente del genoma del nuevo individuo, seleccionan el otro padre de forma independiente.

Los operadores para individuos con codificación real se pueden clasificar en dos grandes grupos:

(i) Cruza discreta: donde para cada componente del genoma se decide aleatoriamente la información de cuál de los dos padres utilizar.

(ii) Cruza intermedia: donde la cría se ubicará, componente a componente, en la media aritmética entre ambos padres. Schwefel propone generalizar la cruza intermedia permitiendo diferentes pesos entre $(0, 1)$ para los padres, es decir, permite alcanzar cualquier posición intermedia entre ambos padres.

Es importante notar que en los casos mencionados anteriormente las crías estarán siempre entre ambos padres, por lo tanto, la cruza puede producir solamente una reducción de volumen cubierto por la población, nunca un aumento.

3.3.2.3. Cruza en los algoritmos genéticos

El operador de cruza es determinante dentro de los algoritmos genéticos, ya que éstos ponen énfasis en esta acción por sobre la mutación. De hecho, muchos algoritmos genéticos no contemplan la mutación de sus individuos sino que sólo la cruza.

Como ya se mencionó, los algoritmos genéticos tradicionalmente utilizan codificación binaria, por lo que aplicarán los operadores de cruza adecuados para dicha codificación.

En los últimos años, para el caso de algoritmos genéticos con codificación real, ha surgido la propuesta de utilizar operadores de cruza centrados en un padre (Ballester & Carter, 2003, 2004; Deb & Agrawal, 1995). Estos consisten en que se generan los nuevos

individuos en el vecindario de uno de los padres, conocido como padre femenino, y donde la dispersión está dada por la distancia entre ambos padres. Basados en este concepto García-Martínez, Lozano, Herrera, Molina y Sánchez (2005) propusieron dividir la población entre padres femeninos y masculinos según su nivel de adaptación.

3.3.2.4. Cruza en las estrategias evolutivas

Contrario a lo que ocurre con los algoritmos genéticos, las estrategias evolutivas no siempre utilizan operadores de cruza al generar sus crías, sin embargo, para aquéllos que sí los utilizan, los parámetros estratégicos, que son parte del genoma de cada individuo, también pueden estar sujetos a cruza y ésta puede ser con otro operador o con otros parámetros que la que afecta a las variables de decisión del problema de optimización.

Schwefel (1977) demuestra que existe una aceleración considerable del proceso de búsqueda para las estrategias evolutivas cuando éstas incorporan la cruza. Adicionalmente, sostiene que la cruza de los parámetros estratégicos es un requisito necesario para facilitar la auto-adaptación de estos mismos (Schwefel, 1987).

3.3.3. Operadores de Mutación

Operador genético asexual, es decir, que utiliza un sólo individuo. Su función es introducir cambios en el individuo inicial para generar un nuevo individuo alterado genéticamente.

En el caso de las estrategias evolutivas el operador más utilizado consiste en cambiar, a través de funciones dependientes de valores aleatorios, los valores de los parámetros estratégicos y luego, con estos nuevos valores, modificar el valor de las variables de decisión. Bäck (1996) presenta estas variaciones de la siguiente manera:

Para la j-ésima variable de decisión, para el i-ésimo individuo en la población donde $N(0,1)$ representa el desarrollo unidimensional de una variable con distribución normal de media 0 y varianza 1, mientras que $N_i(0,1)$ será un vector de i valores obtenidos de una distribución normal de las mismas características.

En primer lugar, para la iteración $t+1$ se modifica la varianza de la mutación m para la i-ésima cría, según la expresión

$$m_i^{t+1} = m_i^t \exp(\tau' N(0,1) + \tau N_i(0,1)) \tag{3.2}$$

con τ y τ' constantes.

A partir de lo anterior, suponiendo y_i y x_i la coordenada i de la cría y el padre respectivamente, se modifican las variables de decisión:

$$y_i = x_i + N(0, \gamma(m_i^{t+1})) \tag{3.3}$$

Es decir, se utiliza el nuevo valor de la varianza para calcular la nueva desviación de la variación introducida a las variables de decisión.

Algunos valores recomendados para las constantes dependen de la dimensión del problema, n.

$$\tau \propto (\sqrt{2\sqrt{n}})^{-1} \tag{3.4}$$

$$\tau' \propto (\sqrt{2n})^{-1} \tag{3.5}$$

En el caso de los algoritmos genéticos, la mutación fue introducida por Holland como un operador secundario, el cual, ocasionalmente modifica algún bit de un individuo que busca cubrir áreas no visitadas. Para esto se define una probabilidad de mutar muy baja, la cual, en caso de cumplirse, cambia el bit correspondiente del individuo por el valor contrario.

3.4. Función de Adaptación

La función de adaptación busca medir la calidad de una solución, para luego poder evaluar cuáles son las mejores soluciones obtenidas hasta el minuto y determinar los individuos sobrevivientes. En general, la función de adaptación corresponde a la evaluación

de un determinado individuo en la función a optimizar. Para el caso de optmización vía simulación este valor, además, no puede conocerse exactamente y estará sujeto al ruido generado por la aleatoriedad del sistema.

Es importante mencionar que el valor de la función de adaptación debe ser siempre positivo e ir aumentando a medida que aumenta la calidad de la solución. Por esto, en el caso de minimización será necesario modificar el tipo de cálculo para que los mejores individuos tengan mejor adaptación.

3.5. Criterio de Término

En la implementación de las estrategias evolutivas se utilizan diferentes condiciones de término, dependiento del criterio del modelador. Entre las más usadas está el realizar una cantidad previamente establecida de iteraciones para el algoritmo y luego obtener el mejor individuo de la población.

Otras opciones son utilizar la diferencia relativa de la adaptación del mejor y el peor de los individuos, fijando una tolerancia para este valor o utilizar la concentración de la población en el espacio solución para continuar iterando hasta que se concentre toda la población en una región de determinado tamaño. En el caso de codificación binaria se puede también estudiar la diversidad del genotipo hasta que se alcance cierto nivel deseado.

3.6. Ranking y Selección

Para el funcionamiento de los algoritmos anteriormente descritos, será fundamental poder seleccionar correctamente los individuos mejor adaptados del sistema. Debido a la aleatoriedad presente en las evaluaciones de cada configuración no se tendrá seguridad del verdadero valor de la función objetivo, por lo que, para disminuir la variabilidad se podrían realizar una gran cantidad de corridas con cada configuración que entregaran mayor seguridad.

Este último mecanismo presenta la dificultad de requerir gran esfuerzo computacional y, por lo tanto, mucho tiempo en la evaluación de adaptación para cada individuo.

La optimización ordinal, presentada por Ho, Sreenivas y Vakili (1992), sostiene que se requiere menor esfuerzo computacional para ordenar soluciones utilizando simulación, que aquel necesario para estimar el mejor valor de la función objetivo. Para esto, una herramienta ampliamente utilizada es la de ranking y selección, cuyo origen es el intento de, a través de la menor cantidad de evaluaciones posibles, asegurar, con cierto nivel de confianza, que se ha seleccionado el mejor sistema de entre un conjunto de posibles configuraciones. Estos consisten en métodos estadísticos desarrollados especificamente para este fin.

Los orígenes de ranking y selección fueron propuestos por Bechhofer (1954), a través de un problema cuyo objetivo es seleccionar la población con mayor media para cierta medida de desempeño entre un set de k poblaciones normales.

Suponiendo que se cuenta con k configuraciones alternativas, o individuos en un algoritmo de búsqueda, para un sistema representados cada uno por $\theta^{[1]}, \theta^{[2]}, ..., \theta^{[k]}$ para el cual se desea maximizar una función objetivo $J(\theta)$ dependiente de alguna variable de salida del sistema. Y_{ij} representa la j-ésima evaluación de la salida para la configuración i, para $i = 1, 2, ..., k$ y $j = 1, 2, ..., n$. Además $\mu_i = E[J(\theta^{[i]})] = E[Y_{ij}]$ representa el valor esperado para el individuo i y $\sigma_i^2 = Var[J(\theta^{[i]})] = Var[Y_{ij}]$ su varianza. Se asume que $\mu_1 \leq \mu_2 \leq ... \leq \mu_k$.

Debido a que cada Y_{ij} son realizaciones de variables aleatorias, es posible que no se seleccione el mejor de los individuos, por lo tanto, se define PSC como la probabilidad de seleccionar al mejor de los candidatos. Sin embargo, podría ocurrir que el modelador fuera indiferente entre seleccionar un individuo u otro si el valor de la función objetivo entre ellos es muy cercana. Para esto se define δ, parámetro de indiferencia, como la mínima distancia que vale la pena detectar. Es decir, si $\mu_k - \mu_{k-1} < \delta$ el modelador será indiferente entre selecionar al individuo k o al $k-1$. Por lo tanto, se busca que se cumpla la siguiente relación para la probabilidad de selección correcta $PSC = P\{\mu_k > \mu_i, \forall i \neq k | \mu_k - \mu_i \geq \delta\} \geq P^*$, donde δ y P^* son definidos por el usuario. Debido a que se puede obtener $PSC = 1/k$

escogiendo aleatoriamente cualquiera de los sistemas con la misma probabilidad, se tiene que $1/k < P^* < 1$.

3.6.1. Métodos de dos etapas

El método original de ranking y selección, propuesto por Bechhofer, asume que cada población tiene una distribución normal con medias desconocidas y diferentes y con varianzas iguales y conocidas. A partir de esto se crea un procedimiento de una etapa, donde se calcula la cantidad de observaciones necesarias para cada diseño alternativo y, luego de calcular cada valor de las esperanzas, se selecciona la alternativa con mayor valor esperado. La cantidad de observaciones queda dada por $N = \lceil (c_{k,P^*}\sigma/\delta)^2 \rceil$, donde los c_{k,P^*} son constantes que se obtienen a partir de una tabla (Bechhofer, 1954) y $\lceil x \rceil$ representa el mínimo entero igual o mayor que x.

Asumir que los μ_i son independientes y que distribuyen normales, es algo común entre estas metodologías y puede apoyarse en la ley de los grandes números cuando los Y_{ij} corresponden al promedio de muchas observaciones durante una simulación. La mayor dificultad de la metodología de Bechhofer es la de asumir varianzas conocidas e iguales. La mayoría de las veces este supuesto no será cierto y, en caso de serlo, será muy dificil conocer su valor a priori.

A partir del caso anterior, las metodologías han evolucionando, pasando en primer lugar a permitir varianzas conocidas y diferentes (Zinger & Nelson, 1958). El primer trabajo donde las varianzas podían ser desconocidas y diferentes fue presentado por Dudewicz y Dalal (1975). Pero el trabajo más conocido y utilizado en este aspecto es la modificación hecha por Rinott (Rinott, 1978). Su trabajo es un mecanismo en dos etapas que consiste en lo siguiente:

(i) Fijar valores para P^* y $\delta > 0$ y la cantidad inicial de observaciones de cada sistema n_0 tal que $n_0 \geq 2$

(ii) A partir de las n_0 muestras para cada configuración calcular la media muestral de cada una a través de:

$$\widehat{\mu}_i^{[1]} = \overline{Y}_i(n_0) = \frac{1}{n_0}\sum_{n_0}^{j=1} Y_{ij} \tag{3.6}$$

Y la varianza según:

$$\widehat{\sigma}_i^2 = S_i^2(n_0) = \frac{1}{n_0 - 1}\sum_{n_0}^{j=1}(Y_{ij} - \widehat{\mu}_i^{[1]})^2 \tag{3.7}$$

Para cada $i = 1, 2, ..., k$.

(iii) Luego se puede calcular la cantidad necesaria de réplicas para cada individuo N_i, la cual está dada por:

$$N_i = \max\left\{n_0, \left\lceil \left(\frac{h\widehat{\sigma}_i^2}{\delta}\right)^2 \right\rceil\right\} \tag{3.8}$$

tal que h es una constante que cumple la relación:

$$P^* = \int_{\infty}^{-\infty} F(t+h)^{k-1} f(t)dt, \tag{3.9}$$

tal que $F(\cdot)$ y $f(\cdot)$ son respectivamente las funciones de distribución y de densidad de una distribución *t de Student* con $n_0 - 1$ grados de libertad. Los valores para h pueden obtenerse de tablas disponibles, donde $h = h(k, P^*, n_0)$.

(iv) Si $n_0 \geq N_i$ no se obtienen más observaciones del sistema i, en caso contrario obtener $N_i - n_0$ observaciones adicionales para cada sistema y calcular la nueva media muestral:

$$\widehat{\mu}_i^{[2]} = \overline{Y}_i(N_i) = \frac{1}{N_i}\sum_{N_i}^{j=1} Y_{ij} \tag{3.10}$$

(v) Finalmente seleccionar el sistema con mayor $\widehat{\mu}_i^{[2]}$ como el mejor.

El método de Rinott es conocido por ser conservador, por lo que, para asegurar la probabilidad de selección correcta (PSC) asume que se encuentra en el caso menos favorable, es decir que $\mu_k - \delta = \mu_{k-1} = ... = \mu_1$.

La principal desventaja de este método es que solamente utiliza la varianza muestral para estimar la cantidad de evaluaciones necesarias para cada escenario. El problema de esto es que pueden existir sistemas que, a partir de la media muestral de la primera etapa, demuestren ser claramente inferiores, información que no se utiliza en este mecanismo, lo que podría llevar a muchas evaluaciones innecesarias del sistema de simulación.

Con el tiempo han surgido trabajos que buscan solucionar esta deficiencia del método de Rinotts. Comenzando con el trabajo de H. Chen, Chen, Lin y Yücesan (1999) y su método conocido como OCBA (*Optimal Computing Budget Allocation*), el cual utiliza información, tanto de la media como de la varianza muestral de la primera etapa, para estimar la cantidad de observaciones para la segunda etapa. El objetivo en este caso es maximizar la probabilidad de éxito dado un presupuesto computacional (cantidad de evaluaciones) máximo fijo.

Hace algunos años E. Chen y Kelton (2000) propusieron una modificación al método de Rinotts para considerar las medias muestrales de la primera etapa en la estimación de la cantidad de evaluaciones requeridas, logrando así evitar evaluaciones innecesarias de escenarios claramente inferiores. La adaptación se produce en el valor de h utilizado para estimar los N_i, el cual cambia para cada individuo quedando como:

$$h_i = \frac{h \cdot \delta}{\max\left\{\delta, \widehat{\mu}_{max}(n_0) - \widehat{\mu}_i(n_0)\right\}} \tag{3.11}$$

con $\widehat{\mu}_{max}(n_0) = \max\left\{\widehat{\mu}_i(n_0)\right\}$ para $i = 1, 2, ..., k$.

Este método es conocido como *enhanced two-stage selection procedure* (ETSS).

Debido a que los h_i son variables aleatorias, no se puede asegurar la probabilidad de selección correcta, por lo que se proponer modificarla para hacer la estimación más conservadora.

Las metodologías presentadas anteriormente son útiles si se utilizan en un conjunto nuevo de escenarios donde se quiere encontrar el mejor de ellos sin contar previamente con observaciones del desempeño de cada uno. Sin embargo, para el caso de los algoritmos evolutivos, se desea encontrar el mejor escenario aprovechando la información que

se pudiera tener de algunos de ellos de iteraciones anteriores. Para esto se tendrá que la cantidad de evaluaciones iniciales para cada escenario será diferente, es decir diferentes n_0.

Para aprovechar al máximo la información disponible, Boesel, Nelson y Kim (2003) presentan una extensión al método de Rinott que permite cantidades diferentes de evaluaciones iniciales para cada individuo. Consiste en lo siguiente:

(i) fijar P^* y definir $n_{min} = \min\{n_{0i}\}$ con n_{0i} la cantidad inicial de evaluaciones para el escenario i. $i = 1, 2, ..., k$.

(ii) fijar $h = h(2, (P^*)^{1/(k-1)}, n_{min})$ desde las tablas para h disponibles.

(iii) determinar la cantidad total de muestras necesarias para cada escenario i, según:

$$N_i = \max\left\{n_{0i}, \left\lceil (\frac{h^2 \cdot S_i^2(n_0)}{\delta^2}) \right\rceil\right\}$$

(iv) obtener $N_i - n_{0i}$ muestras adicionales para cada sistema i.

(v) seleccionar como mejor el sistema con la mayor media muestral obtenida según (3.10).

Existen numerosas variaciones a cada uno de los métodos, así como otras metodologías de realizar ranking y selección. En los últimos años ha aumentado significativamente la bibliografía en esta área, incluyendo el uso de ranking y selección para la optimización vía simulación. Se pueden encontrar numerosas comparaciones y resumenes de los últimos avances (Kim, 2007; Branke, Chick & Schmidt, 2005; Goldsman & Nelson, 1998).

A continuación se presentan algunos avances que han ido penetrando en el uso de ranking y selección en la optimización vía simulación.

3.6.2. Selección y Eliminación

Las metodologías de selección o *screening* buscan eliminar aquellos individuos que sean claramente inferiores a otros de la población. Para esto, se realiza un proceso de selección de subconjunto, el cual devuelve un subconjunto, cuyo tamaño puede ser aleatorio o predeterminado, que continen al mejor de los k sistemas con probabilidad P^* (Bechhofer,

Santner & Goldsman, 1995). Así se evitará realizar un número innecesario de evaluaciones adicionales a estos individuos.

A modo de ejemplo Nelson, Swann, Goldsman y Song (2001) presentan un procedimiento de eliminación para casos en que las varianzas para las muestras de cada sistema son desconocidas y diferentes. Consiste en lo siguiente:

- Determinar nivel de confianza, P^*, parámetro de indiferencia δ, cantidad de muestras $n_0 \geq 2$. Fijar $t = t_{(p^*)^{\frac{1}{k-1}},n_0}$, tal que $t_{\beta,\upsilon}$ representa el cuartil β de una distribución t con υ grados de libertad.
- Obtener Y_{ij}, $i = 1, 2, .., k$ y $j = 1, 2, .., n_0$. Calcular las medias y varianzas muestrales para cada i.
- Para cada par i, p con $i \neq p$ calcular

$$W_{ip} = t \cdot \left(\frac{S_i^2}{n_0} + \frac{S_p^2}{n_0} \right)^{1/2}$$

- Fijar $I = \left\{ i : 1 \leq i \leq k, \overline{Y}_i \geq \overline{Y}_p - \max\left\{0, (W_{ip} - \delta)\right\} \right\}$

I representa el subconjunto seleccionado de individuos sobrevivientes.

Como se puede observar, en este caso no se puede garantizar el tamaño que tendrá el nuevo conjunto I, adicionalmente, I no será nunca un conjunto vacío. En el mejor de los casos I contendrá solamente al mejor de los individuos y, en el peor de los casos, no habrá variado su tamaño inicial.

Boesel et al. (2003) presentan una extensión a este método, que permite diferentes observaciones para cada sistema i. El procedimiento es el mismo, solamente que, en este caso, se definen:

$$t_i = t_{(P^*)^{\frac{1}{k-1}},n_{0i}-1}$$

y

$$W_{ip} = \left(\frac{t_i \cdot S_i^2}{n_{0i}} + \frac{t_p \cdot S_p^2}{n_{0p}} \right)^{1/2}$$

y se fija $I = \left\{ i : 1 \leq i \leq k, \overline{Y}_i \geq \overline{Y}_p - W_{ip} \right\}$

Ninguno de los métodos aquí descritos requieren muestras adicionales de cada sistema más allá de las ya obtenidas. Sin embargo, tampoco pueden garantizar un tamaño fijo para el subconjunto resultante. Este procedimiento puede entregar información muy importante sobre los individuos y la dispersión de sus valores objetivos.

3.6.3. Métodos completamente secuenciales

Los procesos en dos etapas utilizan la primera etapa para estimar la varianza muestral y, a partir de ésta, calcular la cantidad de observaciones necesarias para la comparación. El principal problema de esta aproximación es que la varianza obtenida inicialmente puede ser altamente superior a la varianza real de la población, debido a la cantidad de observaciones con las que se determina, y, por lo tanto, la cantidad de muestras estimadas puede ser mucho mayor a la cantidad realmente necesaria.

Para evitar esta desventaja se puede aumentar el valor de n_0 y así obtener una mejor estimación de los valores para cada sistema alternativo. Esto debe ser hecho de modo muy cuidadoso ya que, si se buscan estimaciones muy precisas, se requerirá una primera etapa muy extensa que podría utilizar incluso más recursos que el método con menor n_0. Para evitar este problema se ha propuesto aumentar la cantidad de etapas, inicialmente a tres. En un principio este número no podía ser mayor, ya que existía un alto costo asociado al cambio de los parámetros de simulación, el cual, hacía difícil realizar más de tres etapas. Gracias a los avances en los computadores, este costo ha ido disminuyendo, restando importancia al cambio de sistemas en la simulación.

Debido a esto, en los últimos años han surgido técnicas completamente secuenciales en las cuales se toma inicialmente un número determinado de observaciones para cada individuo a comparar y, si se cumple determinada condición de término, se obtiene el mejor de ellos o, de ser necesario, se continúa obteniendo cada vez una observación adicional de cada sistema hasta que se cumpla la condición de término. Hartmann (1991) desarrolla y compara métodos secuenciales bajo el supuesto de varianzas desconocidas e iguales.

Se ha probado en diversos trabajos la superioridad de estos métodos frente a los de dos etapas, especialmente cuando se combinan con técnicas de *screening* (Kim & Nelson, 2001; Nelson et al., 2001). Sin embargo, estos procedimientos consideran una cantidad de observaciones inicial igual para cada sistema lo que, como ya se explicó, hará difícil su aplicación al combinarlos con algún algoritmo evolutivo.

Para suplir esta deficiencia Pichitlamken, Nelson y Hong (2006) desarrollan un método completamente secuencial que permite cantidades iniciales diferentes de observaciones para cada individuo, llamado *Sequential Selection with Memory* (SSM). Para su aplicación será necesario almacenar la información sobre cada una de las observaciones obtenidas para cada uno de los individuos.

El procedimiento se clasifica como completamente secuencial con eliminación, es decir, obtiene cada vez una nueva evaluación para cada individuo y elimina aquellas soluciones claramente inferiores lo antes posible. Su aplicación se basa en definir una región de continuidad para la diferencia entre las observaciones de cada par de individuos, si este valor se sale de dicha región uno de los individuos será eliminado de la población.

Si se cuenta con k configuraciones alternativas, Y_{ij} representa la j-ésima evaluación de la salidad para la configuración i, para $i = 1, 2, ..., k$ y $j = 1, 2, ..., n$. El procedimiento SSM consiste en lo siguiente:

- **Parámetros iniciales**: Fijar la probabilidad de selección correcta ($P^* = 1 - \alpha$) y el parámetro de indiferencia (δ). Adicionalmente, n_{0i} representa la cantidad inicial de observaciones disponibles para el individuo i y $n_0 \geq 2$ la cantidad mínima de observaciones para cada sistema.
  Para cada individuo con $n_{0i} < n_0$, obtener $n_0 - n_{0i}$ observaciones adicionales.
- **Estimación varianza**: definir:

$$\underline{n_0} = \min \{n_{0i}\}$$

$$\underline{n_{ip}} = \min \{n_{0i}, n_{0p}\}$$

para todo $i \neq p$.

Para cada par $i \neq p$ se calcula la varianza según:

$$S_{ip}^2 = \frac{1}{\underline{n}_{ip} - 1} \sum_{j=1}^{\underline{n}_{ip}} \left(Y_{ij} - Y_{pj} - \left[\overline{Y}_{i\underline{n}_{ip}} - \overline{Y}_{p\underline{n}_{ip}} \right] \right)^2 \tag{3.12}$$

con

$$\overline{Y}_{in} = \frac{1}{n} \sum_{j=1}^{n} Y_{ij}.$$

Adicionalemente se definen los grados de libertad para cada estimador anterior, según:

$$f_{ip} = \underline{n}_{ip} - 1 \qquad \forall i \neq j$$

- **Parámetros para la región de continuación**: El procedimiento utiliza los siguientes parámetros:

$$\lambda = \tfrac{\delta}{2c}, \quad a_{ip} = \tfrac{\eta f_{ip} S_{ip}^2}{4(\delta - \lambda)}$$

en que η satisface

$$\sum_{l=1}^{c} (-1)^{l+1} \left(1 - \frac{1}{2}\psi(l = c) \right) \left(1 + \frac{(2c - l)l\eta}{2c - 1} \right)^{-f_{ip}/2} = \frac{\alpha}{k - 1} \tag{3.13}$$

donde el indicador $\psi(\epsilon)$ es uno si ϵ es verdadero y cero en caso contrario.

Para $c = 1$, que es el valor recomendado por los autores, se obtiene:

$$\eta = ((k - 1)/(2\alpha))^{2/f_{ip}} - 1. \tag{3.14}$$

- **Definir la región de continuación**: Para lo que se utilizan los parámetros:

$$N_{ip} = \lfloor a_{ip}/\lambda \rfloor$$

$$N_i = \max_{p \neq i} \{N_{ip}\}$$

$$N = \max_{1 \leq i \leq k} N_i.$$

Si $n_0 > N$ entonces parar y seleccionar la solución con mayor $\overline{Y}_{in_{i0}}$ como el mejor. En caso contrario, sea $I = \{1, 2, ..., k\}$ el subconjunto de individuos sobrevivientes, fijar el contador $r = \underline{n}_0$ y $n_{ir} = n_{i0}$ para $1 \leq r \leq n_0$ y continuar en la selección.

- **Eliminación**: Fijar $I^n = I$ y actualizar I según:

$$I = \left\{ i : i \in I^n, r\overline{Y}_{in_{ir}} \geq \max_{p \in I^n p \neq i} (r\overline{Y}_{pn_{pr}-a_{ip}}) + r\lambda \right\}$$

  así I será el conjunto de individuos sobrevivientes.

- **Criterio de término**: si $|I| = 1$, parar y seleccionar al único sobreviviente como el mejor, en caso contrario continuar iterando según:

  – Para cada $i \in I$ tal que $n_{ir} < r + 1$, obtener una observación adicional y fijar $n_{i,r+1} = n_{ir} + 1$.

  – Para $i \in I$ tal que $n_{ir} \geq r + 1$, fijar $n_{i,r+1} = n_{ir}$.

  – Fijar $r = r + 1$. Si $r = N + 1$ terminar el proceso y seleccionar al individuo $i \in I$ con la mayor media muestral como el mejor, en caso contario regresar al paso de eliminación.

Según el procedimiento anterior la región de continuación queda definida acorde a la figura 3.1.

Así, si la sumatoria de la diferencia sobrepasa el límite superior de la región, se elimina el individuo j. Si se abandona la región por el límite inferior se elimina el individuo i. Finalmente, si se sobrepasa el número máximo de iteraciones $r > \left\lfloor \frac{a_{ip}}{\lambda} \right\rfloor$ se elimina el individuo que tenga la menor media muestral.

Una ventaja importante de este método es que puede usarse, no sólo para seleccionar al mejor de los k individuos, sino que también para seleccionar un subgrupo que contenga los m mejores individuos de la población.

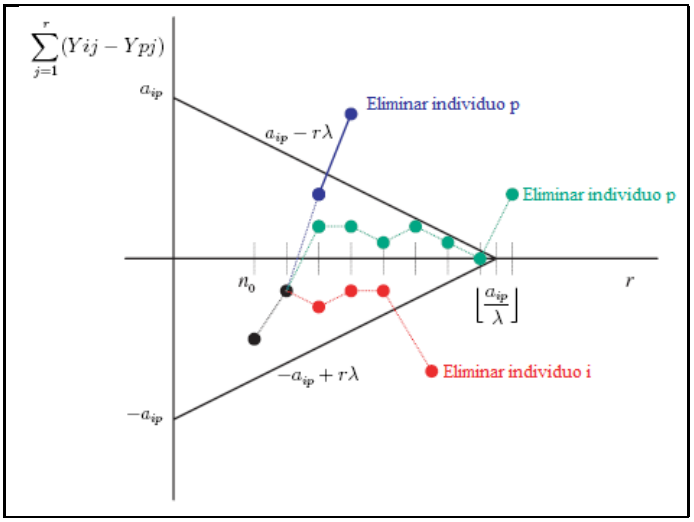

FIGURA 3.1. Región de continuación para método SSM

4. ALGORITMOS UTILIZADOS

En la literatura, existe gran variedad de algoritmos propuestos para la optimización vía simulación. A continuación se presentan los algoritmos seleccionados para probar su desempeño en condiciones de simulación reales. Ambos algortimos pertenecen a la familia de los algoritmos evolutivos, siendo uno de ellos un algorítmo genético, mientras que el otro consiste en una estrategia evolutiva.

4.1. nHGA

Algoritmo genético desarrollado por Olguin (2008) en base al algorítmo genético para optimización de funciones continuas propuesta en Gazmuri y Olguín (2007). Consiste en un algorítmo genético híbrido, es decir, con una etapa de diversificación y una de intensificación con diferenciación genética entre padres femeninos y masculinos y de estado estacionario. El algoritmo genético de estado estacionario es aquel que inserta solamente un nuevo miembro a la población en cada iteración del algoritmo.

En este algoritmo se diferencian dos etapas, la primera, conocida como diversificación, busca recorrer de manera más exhaustiva posible el espacio de solución, mientras que la segunda, conocida como intensificación, centra la búsqueda en una zona prometedora identificada durante la diversificación. Para esto se utilizan dos algoritmos diferentes, propiamente calibrados, para cada tarea.

En este caso, se utilizará el mismo algoritmo en cada una de las etapas pero con diferentes parámetros que cumplan mejor los objetivos de cada una.

La primera etapa comienza con una población de N individuos que son evaluados para obtener su función de adaptación. Entre esta población se seleccionan dos individuos como padres para generar una nueva cría, la cual reemplaza invariablemente al individuo menos adaptado de la población original. El proceso descrito se repite hasta cumplir cierta condición de cambio, después de la cual se da inicio a la etapa de intensificación.

En la etapa de intensificación se repite el mismo proceso pero buscando profundizar y perfeccionar la búsqueda dentro de un valle prometedor detectado en la etapa de diversificación, hasta que se cumpla determinado criterio de término.

A continuación se realiza una descripción detallada de cada parte de este algoritmo.

4.1.1. Población Inicial

Cada individuo será representado a través de su código genético con codificación real, cuyas coordenadas representan diferentes soluciones al problema a optimizar.

El tamaño de la población inicial dependerá de la dimensión del problema, es decir, de la cantidad de variables a optimizar, lo anterior buscando un tamaño de población N que muestree adecuadamente todo el espacio solución pero sin necesitar un esfuerzo computacional demasiado grande para la evaluación de cada individuo. Por lo tanto, si n representa la cantidad de variables a optimizar y según lo propuesto en Olguin (2008), se utilizará un tamaño de población N igual a $15n$.

Para que la población inicial cubra homogéneamente todo el espacio de solución, se propone utilizar el concepto de vecindad, tal como postulan Chelouah y Siarry (2000). Ésta constituye un área de seguridad alrededor de un individuo, dentro de la cual no puede habitar ningún otro individuo de la población. La vecindad consistirá en una ”esfera” de radio ϵ calculado de la siguiente manera:

- Dividir el espacio de búsqueda en N cajas del mismo tamaño, de manera que la suma del volumen de las cajas sea igual al volumen del espacio de búsqueda.
- Definir ϵ como el radio de una esfera inscrita en una de las cajas multiplicado por un factor $n^{1/\sqrt{n}}$, según la siguiente expresión:

$$\epsilon = \frac{n^{1/\sqrt{n}}}{2} \left(\frac{1}{N} \prod_{i=1}^{n} (u_i - l_i) \right)^{1/n} \tag{4.1}$$

  donde u_i y l_i representan el límite inferior y superior de la variable i en el espacio de solución.

En el proceso de creación de la población inicial se genera un individuo aleatoriamente dentro del espacio de solución. Si este nuevo individuo no pertenece a la vecindad de ninguno de los individuos ya creados, se incluye en la población inicial; en caso contrario, se descarta este individuo y se genera uno nuevo. Este proceso se repite hasta generar los N individuos de la población.

4.1.2. Evaluación de adaptación

El algoritmo en discusión está diseñado para minimizar una determinada función objetivo, en consecuencia la función de adaptación debe asignar valores altos de adaptación a individuos con valores de la función objetivo bajos. Por lo anterior, la función de adaptación del individuo i estará dada por:

$$F(i) = \overline{X}_N - \overline{X}_i \tag{4.2}$$

en que $\overline{X}_N$ es el mayor valor entre los $\overline{X}_i$ de toda la población y los promedios se obtienen con todas las muestras disponibles para cada individuo.

Debido a la incertidumbre en los sistemas a simular, no se puede conocer exactamente el valor de la función objetivo al evaluar la simulación según los parámetros de un determinado individuo. Por lo tanto, se pueden cometer errores al definir qué individuo es mejor que otro. Para evitar este problema, pero a la vez evitar el uso innecesario de recursos computacionales, se utilizan técnicas de ranking y selección que permitan garantizar estadísticamente que se está realizando una selección correcta.

Para ordenar adecuadamente los miembros de la población se utilizará una modificación al método SSM, que se muestra en la sección 3.6.3, propuesta por Gazmuri y Olguín (2007), llamado SSM modificado.

La diferenciación sexual entre los individuos de la población, que se ocupará posteriormente en la generación de nuevas crías, debe realizarse según el desempeño de los individuos, logrando identificar aquellos que son claramente mejores dentro de la población para que sean padres femeninos, ya que la cruza centrará los esfuerzos del algoritmo en las zonas donde se encuentren los padres femeninos.

Suponiendo que se pretende identificar el subgrupo de N_F individuos mejor adaptados de la población para que sean los padres femeninos, con una probabilidad aproximada de $P^* = 1 - \alpha$, con un parámetro de indeferencia igual a δ definido por el usuario. Olguin (2008) propone utilizar un valor variable para el parámetro de indiferencia que inicialmente sea mayor a δ y, a medida que el algoritmo progresa, se aproxime hasta alcanzar el valor de δ, debido a que durante las primera iteraciones se pueden encontrar individuos con valores de la función objetivo muy similares, pero que no necesariamente se encuentren cercanos al óptimo y, sin embargo, se requerirá de mayor esfuerzo computacional para determinar cuál de ellos es mejor al utilizar un menor valor de δ. Por lo tanto se utilizará un valor de δ relativo llamado δ'.

El procedimiento utiliza una población de individuos x_i con $i = 1..N$, para cada uno de los cuales se tienen n_0^i evaluaciones de la función objetivo. Cabe recordar que los valores de los n_0^i pueden ser diferentes entre ellos e, incluso cero, para los nuevos individuos de la población. El método SSM modificado será el siguiente:

(i) Definir los parámetros de entrada P^*, como la probabilidad de selección correcta, δ, el valor del parámetro de indiferencia y n_0, la cantidad mínima de evaluaciones para cada individuo de la población.

(ii) Definir I como el conjunto de los individuos de la población. Para cada individuo en I que tenga un número de muestras iniciales n_0^i menor a n_0, realizar $n_0 - n_0^i$ réplicas adicionales.

(iii) Inicializar los valores de las variables que contienen la mínima cantidad de réplicas actuales (r) y la cantidad de muestras disponibles para cada individuo ($n_{i,r}$), tales que:

$$r = \min_{i \in I} \left\{ n_0^i \right\} \tag{4.3}$$

$$n_{i,r} = n_0^i \tag{4.4}$$

Adicionalmente definir el parámetro de indiferencia relativo como:

$$\delta' = \max\left\{\delta, \frac{N_F}{N}\Delta\right\} \tag{4.5}$$

donde Δ representa la diferencia en el valor de la función objetivo entre el mejor y el peor de los individuos de la población, es decir:

$$\Delta = \max_{i\in I}\left\{\bar{X}_i(n_{i,r})\right\} - \min_{j\in I}\left\{\bar{X}_j(n_{j,r})\right\} \tag{4.6}$$

(iv) Calcular las variables del método según las observaciones a partir de las siguientes expresiones:

$$\underline{n}_{ij} = \min\left\{n_{i,r}, n_{j,r}\right\} \qquad \forall\, i \neq j \tag{4.7}$$

$$f_{ij} = \underline{n}_{ij} - 1 \qquad \forall\, i \neq j \tag{4.8}$$

y estimar las varianzas $\sigma_{ij}^2 = Var\left\{f(x^{(i)}) - f(x^{(j)})\right\}$ según la expresión

$$S_{ij}^2 = \frac{1}{\underline{n}_{ij} - 1}\sum_{p=1}^{\underline{n}_{ij}}\left(X_{i,p} - X_{j,p} - \frac{1}{\underline{n}_{ij}}\sum_{q=1}^{\underline{n}_{ij}}\left(X_{i,q} - X_{j,q}\right)\right)^2 \qquad \forall\, i \neq j \tag{4.9}$$

con $X_{i,p}$ el valor de la función objetivo del individuo i en la p-ésima réplica del modelo de simulación.

(v) Definir la región de continuación, área según la cual se eliminarán los individuos cuyo valor de la función objetivo excede al valor del mejor individuo más una cierta tolerancia, según los siguientes parámetros:

$$\lambda = \frac{\delta'}{2} \tag{4.10}$$

$$a_{ij} = \frac{\eta f_{ij} S_{ij}^2}{4\left(\delta' - \lambda\right)} \tag{4.11}$$

tal que

$$\eta = \left(\frac{|I| - 1}{2\alpha}\right)^{\frac{2}{f_{ij}}} - 1 \tag{4.12}$$

Y definir también los siguientes parámetros:

$$N_{ij} = \left\lfloor \frac{a_{ij}}{\lambda} \right\rfloor \qquad \forall\, i \neq j \tag{4.13}$$

$$N_i = \max_{i \neq j} \{N_{ij}\} \qquad \forall\, i \in I \tag{4.14}$$

$$N_{max} = \max_{i \in I} \{N_i\} \tag{4.15}$$

si $r > N_{max}$ pasar al paso (viii)

(vi) Luego de definir $I' = I$, la siguiente expresión resume la etapa de eliminación a través de la actualización del conjunto I:

$$I = \left\{ i : i \in I' \;\; y \;\; r\overline{X}_i(n_{i,r}) \leq \min_{j \in I', j \neq i} \left(r\overline{X}_j(n_{j,r}) + \alpha_{ij} \right) - r\lambda \right\} \tag{4.16}$$

(vii) Si $|I| \leq N_F$ pasar al paso (viii). En caso contrario realizar los siguientes procesos y luego volver al paso (iii):

- Para cada individuo i aún vigente en el conjunto I,si $n_{i,r} < r + 1$ realizar una réplica adicional y definir $n_{i,r+1} = n_{i,r} + 1$.
- Definir $n_{i,r+1} = n_{i,r}$ para cada individuo que ya no pertenece al conjunto I y para cada individuo i aún vigente y en los que $n_{i,r} \geq r + 1$.
- Definir $r' = r + 1$ y actualizar las siguientes variables:

$$r = \min_{i \in I} \{n_{i,r'}\} \tag{4.17}$$

$$n_{i,r} = n_{i,r'} \forall i \in \{1, ..., N\} \tag{4.18}$$

(viii) *Finalización*: renumerar los N individuos de la población en orden creciente según su valor promedio de la función objetivo, de tal modo que $x^{(1)}$ será el individuo con menor valor promedio de la función objetivo.

La diferencia del método SSM con el método SSM modificado es que, en el primero, la región de continuación queda definida según los parámetros iniciales, sin aprovechar la mejora en estimaciones de elementos como las varianzas que pudieran obtenerse a medida que se avanza en el desarrollo, mientras que el segundo actualiza cada vez los parámetros

y, de esta manera, la región de continuación, haciéndolo más eficiente, como se puede ver según pruebas empíricas en el trabajo de Olguin (2008).

4.1.3. Generación de Crías

Este algoritmo genera una sola cría en cada iteración. Para esto se seleccionan dos padres diferenciados sexualmente, el padre femenino es alrededor del cual se realizará la exploración del espacio solución, ya que éste fue seleccionado entre los N_F individuos mejor adaptados, mientras que el padre masculino será utilizado según su distancia con el padre femenino, la cual definirá la dispersión de la cría respecto al padre feminino.

Por lo anterior, se utilizarán diferentes mecanismos de selección para cada uno de los padres, de modo de cumplir los diferentes objetivos de cada uno.

Para la selección del padre femenino, será importante seleccionar un individuo bien adaptado, por lo tanto, se utilizará el método de rueda de ruleta descrito en 3.3.1.2, donde a cada individuo, entre el grupo de los N_F individuos mejor adaptados, se le asigna una probabilidad de ser elegido proporcional al valor de su función de adaptación, favoreciendo así la elección de individuos mejor adaptados.

En el caso del padre masculino, se desea seleccionar un padre que se encuentre lejos del padre femenino previamente seleccionado. Para esto se utiliza el método de selección de emparejamiento variado negativo, el cual escoge en primer lugar, aleatoriamiente entre los N_M individuos mejor adaptados, un grupo de N_C candidatos y selecciona como padre el que tenga la mayor distancia euclidiana al padre femenino. En este caso se utiliza un valor de $N_C = 5$.

Los padres seleccionados generarán una nueva cría utilizando el método de cruza PNX: Denominando x_i^F la componente i del padre femenino, x_i^M la componente i del padre masculino y x_i^C la componente i de la nueva cría, se tiene que esta última estará definida según:

$$x_i^C = N\left(x_i^F, \frac{\left|x_i^F - x_i^M\right|}{\chi}\right) \tag{4.19}$$

donde $N(\mu, \sigma)$ representa un número aleatorio obtenido de una distribución normal con media μ y desviación estándar σ. χ será un parámetro ajustable del método, según el cual, mientras mayor sea su valor, más concentrada será la busqueda alrededor del padre femenino.

La estrategia de reemplazo con la nueva cría consistirá en reemplazar al peor individuo de la población con este nuevo individuo, lo cual evita evaluar su función objetivo antes de incorporarlo a la población.

4.1.4. Etapa de Diversificación

En las secciones anteriores se explicaron las caracterísiticas generales del algoritmo genético utilizado en cada una de las etapas. La etapa de diversificación buscar cubrir de manera extensiva el espacio de solución y poder identificar las zonas prometedoras donde podría encontrarse el óptimo del problema. Para cumplir este objetivo se variarán algunos parámetros del método según como se propone en Olguin (2008).

Los primeros valores serán los de N_F y N_M, acorde con el objetivo descrito, se seleccionaron estos valores en $N_F = 0,5 \cdot N$ y $N_M = N$. Además, para la selección del padre masculino, se utilizará un valor de $N_C = 5$ y el valor de χ se fijará en 2.

Para dar por concluida la etapa de diversificación, se utilizarán dos criterios independientes. El primero consiste en alcanzar cierto número máximo permitido de soluciones visitadas para esta etapa, fijado en $50n^2$, mientras que el segundo consistirá en que todos los padres femeninos estén contenidos en un área prometedora, es decir, que se cumpla la condición:

$$\frac{1}{N_F} \sum_{i=2}^{N_F} \left\| x^{(i)} - x^{(1)} \right\| \leq \epsilon_r \tag{4.20}$$

con $\|...\|$ la distancia euclidiana y ϵ_r tal como se define en la ecuación (4.1).

4.1.5. Etapa de Intensificación

La etapa de intensificación busca explorar la zona prometedora identificada y aproximarse cada vez más a la mejor solución del problema. Para cumplir este objetivo se fijan los siguientes parámetros del problema $N_F = 0,1 \cdot N$, $N_M = 0,5 \cdot N$, $N_C = 5$ y $\chi = 4$.

Como criterio de término para el algoritmo se utilizarán dos criterios independientes, de manera similar a la etapa de diversificación. El primero de ellos será alcanzar un número máximo permitido de soluciones visitadas en la etapa de intensificación, definido como $50n^2$, mientras que el segundo se cumple cuando todos los individuos están los suficientemente aglomerados. Matemáticamente esto se define cuando:

$$\frac{1}{N}\sum_{i=2}^{N}\left\|x^{(i)} - x^{(1)}\right\| \leq 0,5\epsilon_r \tag{4.21}$$

Una vez cumplida cualquiera de las dos condiciones anteriores, se selecciona como solución para el problema de optimización el individuo con mayor valor de su función de adaptación.

4.2. ISS

Estrategia evolutiva propuesta por Buchhloz y Thümmler (2005), en la cual se incorporan procedimientos estadísticos para seleccionar eficientemente el mejor de los individuos, donde este último se define como aquel con mayor valor esperado en la función objetivo. En este caso particular se utiliza una estrategia del tipo $\mu + \lambda$-ES pero podría utilizarse, sin perder generalidad, una estrategia μ, λ-ES.

Este algoritmo innova además utilizando una población de elite, la cual tiene un tamaño ω predefinido por el usuario. Esta población será la encargada de almacenar, a lo más, los ω individuos mejor adaptados encontrados hasta el momento.

La notación a utilizar será la siguiente:

- P^t corresponde a la población de posibles padres disponibles para la iteración t.
- Q^t designa la población de λ nuevos individuos (crías) de la iteración t.

- β^t indica el conjunto de población elite descrito anteriormente, cuyo tamaño será como máximo de ω individuos.

Siguiendo la lógica de las estrategias evolutivas,en términos generales, el algoritmo propuesto por Buchhloz y Thümmler (2005) consiste en lo siguiente:

(i) Definición de Parámetros: En primer lugar se fijan los parámetros y se define el espacio solución factible para las variables de decisión.

(ii) Inicialización Población: A continuación se inicializa la población de padres P y la población elite β.

(iii) Proceso Iterativo: A partir de este punto se realiza un proceso iterativo hasta que se cumpla determinada condición de término establecida por el usuario. Los siguientes son los pasos del proceso iterativo.

- Selección: Escoger λ individuos entre la población de padres a través de una selección aleatoria uniforme.
- Generación Crías: A través de operadores genéticos los individuos escogidos mutan para crear las nuevas crías. En caso que una cría no esté dentro del espacio solución, se repetirá la mutación hasta que dicha condición se cumpla.
- Evaluación Adaptación: Todos los individuos, conjunto de padres e hijos, son evaluados de acuerdo a una estrategia de selección, la cual indicará la cantidad de muestras necesarias para cada individuos, sin embargo, la selección se llevará a cabo escogiendo entre todo el conjunto los individuos con mayor valor de su función de adaptación.
- Sobrevivencia: Se seleccionan los μ individuos con mayor valor de la función de adaptación, los cuales conformarán la nueva población P^{t+1}.
- Población Elite: Actualización de la población de elite β con los individuos mejor adaptados encontrados hasta el momento.

(iv) Selección Final: Una vez cumplida la condición de término se realiza un nuevo proceso de selección estadistica para poder identificar al mejor individuo en la población elite.

A continuación se detallan las principales características de este algoritmo.

4.2.1. Población Inicial

La representación de cada individuo se realiza utilizando un genoma con codificación real. Este genoma, además de indicar las coordenadas del individuo dentro del espacio solución, contiene información sobre la mutación del individuo, conocida como fuerza de mutación.

La fuerza de mutación representa la magnitud de la varianza para cada coordenada del espacio solución. La ventaja de incorporar esta característica en la codificiación genética de cada individuo es que ésta irá mutando a través de las generaciones, permitiendo a cada individuo adaptarse más rápidamente.

El tamaño de la población debe ser determinado por el usuario, lo que requerirá mayor conocimiento del espacio solución para evitar los problemas de sobremuestrear el espacio o de dejar grandes áreas sin visitar.

Una vez determinado el tamaño de la población, ésta se crea aleatoriamente, según una distribución uniforme en el espacio solución. Contrario a lo que ocurre en el método anterior, en este caso cada nuevo individuo es totalmente independiente a los generados previamente.

4.2.2. Generación de Crías

Según la notación utilizada, μ representa el tamaño de la población, definido por el usuario, mientras que λ representa la cantidad de nuevos individuos (crías) que se generan en cada iteración.

El proceso de generación de crías sigue la lógica de las estrategias evolutivas. Entre la población de posibles padres P^t, de tamaño μ, sobreviviente de la iteración anterior se

seleccionan aleatoriamente λ padres, cada uno encargado de generar una nueva cría. La selección se realiza asignando a cada individuo igual probabilidad de ser elegido como padre, independientemente del valor de la función de adaptación, es decir, los padres se seleccionan uniformenmente entre toda la población. Por lo tanto se utiliza el mecanismo de Selección Aleatoria descrito en la sección 3.3.1.

Durante el proceso de mutación se ajusta, en primer lugar, el valor de la fuerza de mutación, correspondiente a la desviación estándar de la distribución normal utilizada para la mutación. Sea m_{ij} el factor de mutación del padre i para la variable j, el factor de mutación $\tilde{m}_{ij}$ para la i-ésima cría será determinado por:

$$\tilde{m}_{ij} = m_{ij} \cdot \exp(u_i/\sqrt{2n} + u_{ij}/\sqrt{2\sqrt{n}}) \tag{4.22}$$

donde los u_i y u_{ij} corresponden a desarrollos de una distribución normal con media 0 y varianza 1, y n representa la cantidad de variables a optimizar.

Esta expresión corresponde a la presentada en (3.2) con τ y τ' como se muestran en las ecuaciones (3.4) y (3.5).

El siguiente paso corresponde a la mutación de las variables de decisión. Sea x_{ij} el valor para el factor j en el individuo i del grupo de padres y $\tilde{x}_{ij}$ el valor correspondiente a la respectiva cría. Se tiene que las coordenadas del nuevo individuo estarán dadas por la expresión:

$$\tilde{x}_{ij} = x_{ij} + \tilde{m}_{ij} \cdot u_{ij} \tag{4.23}$$

con los u_{ij} obtenidos a partir de una distribución $N(0, 1)$.

Dado que este algoritmo no contempla la cruza entre padres, con el proceso de selección y mutación anteriormente descrito quedan definidas las crías para la presente iteración.

4.2.3. Evaluación de Adaptación

Este algoritmo busca maximizar la función objetivo, por lo tanto, en este caso se designa como adaptación de un individuo al valor de la función objetivo al utilizar la configuración indicada por su genoma. Dicho valor tiene como objetivo final lograr diferenciar aquellos miembros de la población que tiene un mejor valor de la función objetivo para que sean los sobrevivientes para la siguiente iteración, es decir, para configurar la población P^{t+1}.

Sean X_{ij} el valor de la j-ésima evaluación del modelo de simulación para la configuración del individuo i y n_{0i} la cantidad de evaluaciones obtenidas para la configuración i. Entonces, la función de adaptación f_i para el individuo i será:

$$f_i = \frac{1}{n_{0i}} \sum_{j=0}^{n_{0i}} X_{ij} \tag{4.24}$$

Adicionalmente, dicho valor servirá, al finalizar el algoritmo, para determinar la mejor solución encontrada por el procedimiento.

Ya que el valor de la función de adaptación no podrá ser conocido con exactitud y deberá ser estimado a partir de las evaluaciones del sistema de simulación correspondiente, será necesario lograr tener la mayor exactitud posible en la adaptación de cada individuo, buscando además, minimizar la cantidad de evaluaciones de simulación requeridas para cada posible configuración del sistema.

Debido a que, inicialmente, sólo se desea determinar cuáles son los individuos mejor adaptados de la población, será necesario poder ordenar los individuos más que conocer con exactitud el valor de su función de adaptación, lo cual requiere un menor uso de recursos.

Para lo anterior se utiliza la siguiente técnica de Ranking y Selección.

4.2.3.1. Ranking y Selección

El trabajo desarrollado por Buchhloz y Thümmler (2005) propone utilizar diferentes métodos de selección estadísticos para la selección de los μ individuos mejor adaptados.

En el presente trabajo se utilizará exclusivamente una de ellas que demuestra tener buenos resultados en las pruebas presentadas.

La metodología seleccionada utiliza Ranking y Selección, para lo cual el usuario debe definir sus parámetros, es decir, debe fijar: P^*, la probabilidad de selección correcta y el parámetro de indiferencia, δ, como la mínima distancia que vale la pena detectar, es decir, el usuario será indiferente entre escoger alguno de dos individuos cuyas funciones objetivos estén a una distancia menor a δ.

En primer lugar se utiliza un procedimiento de pre-selección tal como se propone en Bechhofer et al. (1995) para eliminar a los individuos claramente inferiores. El método utilizado es el propuesto por Boesel et al. (2003), tal como se muestra en la sección 3.6.2, conocido como *Extended screen-to-the-best procedure*. Éste fija las condiciones bajo las cuales un individuo puede ser eliminado de la población mientras se busca el conjunto con los k miembros mejor adaptados. Sus ventajas son que utiliza diferentes cantidades de observaciones iniciales para cada miembro de la población y no requiere muestras adicionales para cada individuo pero que, por lo mismo, no puede garantizar cierto tamaño en la población final.

En combinación con lo anterior se usa un mecanismo iterativo de selección conocido como *iterative subset selection procedure (ISS)*, el cual busca reducir un set de tamaño H a un conjunto de tamaño máximo m definido por el usuario. Este mecanismo consiste en continuar obteniendo nuevas observaciones de cada individuo sobreviviente hasta que el conjunto alcance el tamaño m predefinido, para lo cual utiliza recurrentemente el método *Screen-to-the-best*. Debido a lo anterior, no se puede garantizar que se cumpla la condición de que $P\left\{k \in H : \mu_k - \mu_{k-1} \geq \delta\right\} \geq P^*$.

El método anteriormente enunciado, que toma una población H con k individuos y entrega una población con un tamaño de a lo más m, consiste en los siguientes pasos:

(i) Definir los parámetros de Ranking y Slección, es decir, la probabilidad de selección correcta P^* y el parámetro de indiferencia δ.

(ii) Calcular la probabilidad ajustada $P^{II} = P^{*1/(k-m)}$.

(iii) Fijar $n_0 = \min_{i=1..k} n_{0i}$ con n_{0i} el número de muestras conocidas para cada individuo i.

(iv) Obtener $n_0 - n_{0i}$ muestras de cada individuo i.

(v) Ejecutar procedimiento Screen-to-the-best con el conjunto H y los parámetros P^*,$\delta/2$ y n_{0i} muestras.

(vi) Fijar $n_0 = n_0 + 1$.

(vii) Si el tamaño de la población H es mayor que el tamaño deseado m, es decir si $|H| > m$, volver al paso (iii).

El algoritmo Extended Screen-to-the-best consiste en lo siguiente:

(i) Para cada individuo i de la población H obtener la media y varianza de las muestras de cada individuo:

$$\bar{X}_i = \frac{1}{n_{0i}} \sum_{j=0}^{n_{0i}} X_{ij} \quad , \quad S_i^2 = \frac{1}{n_{0i}-1} \sum_{j=0}^{n_{0i}} (X_{ij} - \bar{X}_i)^2$$

(ii) Fijar t_i como el $(P^*)^{1/(k-1)}$ cuartil de una distribución t con $n_{0i} - 1$ grados de libertad.

(iii) Para cada par de individuos i, j dentro de la población, calcular los pesos:

$$W_{ij} = \sqrt{\frac{t_i^2 S_i^2}{n_{0i}} + \frac{t_j^2 S_j^2}{n_{0j}}}$$

Si $X_i < X_j - \max\{0, W_{ij} - \delta\}$ entonces se elimina al individuo i de la población H.

(iv) Entrega como salida la nueva población H.

Una vez finalizado el procedimiento ISS se continúa con la selección de la población sobreviviente.

4.2.4. Población Sobreviviente

El mecanismo anteriormente descrito servirá para cononcer la cantidad de evaluaciones necesarias para cada individuo de la población. Una vez realizado éste, se selecciona como

población sobreviviente a los μ individuos con mayor valor en su función de adaptación. Esta nueva población constituirá los posibles padres para la siguiente etapa.

La población elite (β) también será actualizada en cada iteración. Para lo anterior se realiza una vez el proceso de *Screen-To-The-best* con el conjunto $\beta \cup Q \cup P$ y $\delta = 0$ y se conforma la población elite con $\max\{\omega, |H|\}$ individuos con mejor valor en su función de adaptación entre $\beta \cup Q \cup P$.

4.2.5. Criterio de Parada

El algoritmo continúa iterando hasta que se cumple cierta condición establecida por el usuario. La condición de término a utilizar será cuando se alcance un determinado número de iteraciones previamente definidas.

Finalmente, entre los individuos de la población de elite se seleccionará el mejor de los individuos, utilizando nuevamente el criterio de ranking y selección con $m = 1$ y $\delta = 0$. Así, se escogerá como solución al mejor individuo encontrado por el algoritmo.

4.3. OptQuest

OptQuest, software de optimización via simulación desarrollado por OptTek System Inc. Es uno de los softwares de su tipo más utilizados en la actualidad, Fu et al. (2005) menciona algunas de las plataformas de simulación que lo utilizan como herramienta de optimización, entre las que se encuentran Arena, Crystal Ball, ProModel y SIMUL8.

El software fue desarrollado inicialmente por F. Glover, J.P. Kelly y M. Laguna en la Universidad de Colorado con la finalidad de optimizar modelos de simulación discreta modelados con Micro Saint 2.0.

Su funcionamiento exacto no es público, pero combina metaheuristicas de búsqueda Tabú, redes neuronales y scatter search en una sola heurística de optmización (Kleijnen & Wan, 2007). Por lo tanto, utiliza el sistema de simulación como una caja negra encargada de evaluar la función objetivo en cada configuración para las variables de decisión.

April, Glover, Kelly y Laguna (2003) presenta algunas indicaciones sobre el funcionamiento del software. El cual es similar al de un algoritmo evolutivo, utilizando metodologías de búsqueda tabú combinadas con scatter search para crear las crías de una determinada población. Adicionalmente, utiliza una metodología para filtrar las soluciones inferiores respecto a la mejor solución encontrada hasta el momento. Para esto utiliza redes neuronales para construir un metamodelo del espacio solución y, a continuación, aplica determinadas reglas para eliminar aquellas soluciones claramente inferiores, ahorrando así parte del esfuerzo computacional utilizado para realizar simulaciones que no representarán un avance para el proceso.

Para obtener un metamodelo que sea una buena representación del problema, OptQuest debe decidir cuándo las redes neuronales han recolectado suficiente información para comenzar a aplicar los filtros.

Entre los parámetros que debe decidir el usuario para utilizar OptQuest se cuentan los límites superiores e inferiores para las variables de decisión, las cuales pueden ser discretas o continuas, además de utilizar valores sugeridos de las mismas para utilizarlos como punto de partida. Según lo que se ha demostrado, su eficiencia depende altamente de los valores sugeridos.

Una de las ventajas de este software es que permite manejar una amplia gama de restricciones para las variables de decisión. En primer lugar permite todo tipo de restricciones sobre las variables de decisión, tanto lineales como no lineales, según las cuales, al crear un nuevo individuo, evalúa su factibilidad y, de no cumplir con alguna de ellas adapta al nuevo individuo para cumplir con la restricción. En segundo lugar, permite variables que dependerán de respuestas del sistema de simulación, para su manejo evalúa en primer lugar el sistema y utiliza técnicas estadísticas, según las cuales se puede definir un criterio de precisión, para determinar si un individuo cumple o no con dichas restricciones.

Como criterio de precisión, tanto para la función objetivo como para la evaluación de restricciones que dependan de respuestas del sistema, el usuario puede escoger entre dos opciones:

(i) Especificar un número definido de réplicas para cada nuevo individuo.

(ii) Permitir que el programa se mueva dentro de un rango donde se especifica el mínimo y máximo de réplicas permitidas, según las cuales se detendrá si encuentra una solución claramente inferior, para lo cual utiliza un test de hipótesis nulo de Student.

Finalmente, el usuario debe especificar un criterio de parada para la optimización. Usualmente se utiliza un tiempo determinado o se detiene la optimización cuando se han realizado un cierto número de iteraciones sin mejorar el valor objetivo.

Existe variada literatura sobre usos y aplicaciones de OptQuest para diversos problemas, especialmente para optimización de portafolios de inversión, diferentes problemas de producción y problemas de optimización determinística tanto lineales como no lineales, entre ellos se encuentran (Laguna, 1997; Kleijnen & Wan, 2007; Grewal, Rogers & Enns, 2008; Rogers, 2002).

5. SISTEMAS DE SIMULACIÓN

Con el propósito de probar el funcionamiento de los algoritmos anteriormente descritos, se realizó una revisión bibliográfica y selección de sistemas de simulación que pudieran cumplir los objetivos.

Los sistemas seleccionados cumplen con poseer variables de decisión continuas y ser capaces de representar problemas reales en los que se podría desear utilizar la optimización vía simulación. Cada uno de ellos fue programado a través del software de simulación Arena.

5.1. Sistema 1: Abastecimiento de Combustible Ejército Turco

El primer sistema a optimizar se basa en el trabajo presentado por Sabuncouglu y Hatip (2005), donde se describe la aplicación de simulación en el sistema de abastecimiento de combustible del ejército turco. Un problema que podría ser crítico en caso de entrar en guerra debido a la necesidad de combustible para movilizar las tropas, con la dificultad de lograr abastecerlas a todas en un territorio extenso.

Debido a la confidencialidad con ciertos datos del trabajo, Sabuncouglu y Hatip (2005) entrega una descripción general del sistema, por lo que ciertos parámetros debieron ser fijados arbitrariamente.

5.1.1. Descripción del Sistema

El sistema de abastecimiento consiste en barcos para el traslado marítimo y traslado y almacenamiento terrestre por medio de cañerías y depósitos. Existen tres tipos diferentes de combustible, jet, fuel y diesel, las cuales no se mezclan en ningún momento.

El ejército turco cuenta con catorce estaciones de acopio distribuidas en su territorio, las cuales son las encargadas de satisfacer las diferentes demandas. Cada estación posee tanques para diferentes tipos de combustible, con capacidades definidas en cada uno de ellos.

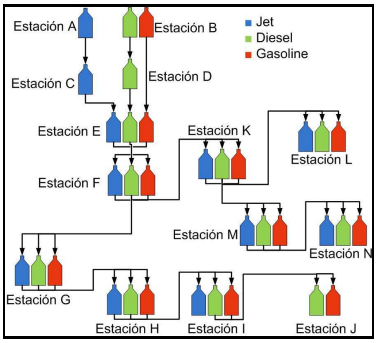

FIGURA 5.1. Configuración del sistema de tanques de combustible.

La figura 5.1 muestra la configuración de las estaciones terrestres con los tanques en cada una de ellas y las cañerías de comunicación. Se asume que el sistema tiene suficiente combustible debido al tamaño de la refinería, es decir, siempre que se requiera nuevo combustible para ingresar al sistema, éste estará disponible.

Para el transporte desde la refinería hasta el conjunto de tanques los militares utilizan dos tipos de navíos, los tipo X, que pueden transportar 7.000 metros cúbicos de combustible y los tipo Y con una capacidad de 5.000 metros cúbicos. Estos navíos llegan hasta las estaciones A o B dependiendo el tipo de combustible que transportan.

La capacidad de almacenaje para los diferentes tanques varía entre los 2.500 y los 45.000 metros cúbicos, mientras que las cañerías también varían en su capacidad de transporte (mayor detalle sobre el sistema y sus características se puede encontrar en los anexos). Las cañerías pueden transportar cualquiera de los tres tipos de combustible sin necesidad de limpieza, sin embargo, sólo pueden contener un tipo de combustible a la vez.

El combustible en las estaciones A y B es transportado hacia las siguientes estaciones lo más rápidamente posible, para poder poner una nueva orden a la refinería. Los niveles para poner una nueva orden en las estaciones iniciales son de 38.000 metros cúbicos en la estación A de jet y 8.000 metros cúbicos en cada uno de los tanques de gasolina y diesel en la estación B. Cuando se pone una nueva orden, la refinería la asigna a los transportes disponibles en ese momento.

Las estaciones C y D vacían su contenido hacia las estaciones siguientes continuamente para poder recibir el combustible proveniente de las estaciones A y B.

El flujo entre las siguientes estaciones se determina de la siguiente manera:

- En la estación receptora se revisa el contenido de cada uno de los tanques, aquel que tenga un menor contenido porcentual será el que solicite un lote de combustible a la estación de origen.
- En la estación de origen se revisa el contenido del tanque correspondiente, si éste tiene sufiente combustible entonces se realiza el envío, en caso de no tener disponiblidad se realiza el mismo procedimiento con los dos combustibles restantes.
- Si la cantidad de combustible en el destino es superior al 95%, entonces no se realiza la orden.

Se considera que un tanque en la estación de origen tiene suficiente combustible si tiene un contenido mayor a un determinado nivel porcentual identificado como nivel mínimo. Si durante el envío el contenido del tanque de origen cae bajo su nivel mínimo, entonces se interrumpe el envío.

Para el caso de la demanda por combustible se realizan los siguientes supuestos:

- El tipo de combustible más demandado es jet, luego viene diesel y finalmente la gasolina, que tiene una demana muy baja.
- La llegada de una nueva demanda a cada estación distribuye exponencial con diferente tasa en cada una.
- De cada nueva demanda, a partir de probabilidades definidas se determina qué tipo de combustible es el que requiere.
- La cantidad de combustible demandada sigue una distribución triangular en cada caso.

La simulación termina cuando algún tanque no puede satisfacer la demanda recibida.

5.1.2. Problema de Optimización

El objetivo del problema de optimización es buscar los niveles mínimos para cada tanque de modo de maximizar el tiempo en el cual el sistema puede satisfacer la demanda.

Así, el problema contará con 20 variables de decisión que representan el porcentaje mínimo del contenido de cada tanque para que este pueda entregar combustible a las siguientes estaciones, de acuerdo al mecanismo ya descrito, por lo tanto, todas las variables estarán entre 0 y 100.

Se modeló en el software Arena el sistema anteriormente descrito. El uso de este software, diseñado específicamente para simulación, permite al usuario tratar gran cantidad de problemas que, en caso de utilizarse un software de propósito general, dificultarían la modelación y validación de los sistemas.

Para la validación del sistema de simulación se utilizó, en primer lugar, las animaciones disponibles en Arena, así, se pudo observar la evolución de los niveles de combustible durante el proceso para ver que éste se comportara según lo deseado. Adicionalmente, a modo de ejercicio, se estudió la función objetivo del problema de optimización suponiendo que el nivel mínimo debe ser el mismo para todos los tanques.

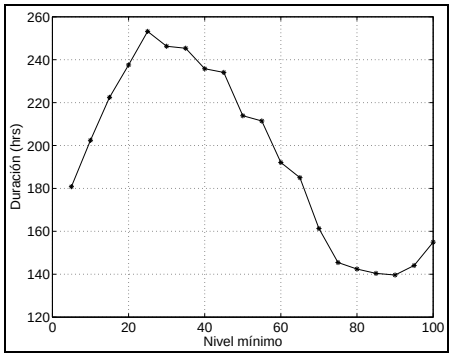

FIGURA 5.2. Resultados con los niveles mínimos iguales para todos los tanques

La figura 5.2 muestra los resultado obtenidos con el ejercicio mencionado. En ella se puede observar que, para niveles mínimos iguales en todos los tanques, el sistema alcanza su mejor desempeño con un nivel mínimo de 25%. Adicionalmente, la forma de la curva sirve para validar el funcionamiento del modelo de simulación donde, dependiendo de los

niveles mínimos, cambiará el tiempo durante el cual el sistema es capaz de satisfacer la demanda.

5.2. Sistema 2: Mantención de Aviones de la Flota Finlandesa

El siguiente sistema a optimizar se basa en el trabajo presentado por Ville y Kai (2008), el cual presenta el sistema de simulación desarrollado para el estudio de las políticas de mantención de los aviones de la Fuerza Aéra Finlandesa. En él se estudian principalmente los diferentes procedimientos de mantención por los que debe pasar cada avión de acuerdo a las políticas establecidas.

5.2.1. Descripción del Sistema

El siguiente modelo se basa en el trabajo anteriormente mencionado, sin embargo, debido a la confidencialidad de la información y a los objetivos del sistema a simular, ha sido adaptado según las nuevas necesidades.

La Fuerza Aérea cuenta con una flota determinada de aviones, distribuida en tres unidades primarias de operaciones, conocidas como comandos aéreos. En cada uno de ellos un escuadrón de combate es el responsable de la operación de los vuelos y de ciertas actividades de mantenimiento.

Cada comando aéreo cuenta, adicionalmente, con un taller mecánico donde se llevan a cabo labores de reparación y mantenimiento más avanzadas. Finalmente, la Fuerza Aérea cuenta con un almacén nacional, donde se realizan las actividades más complejas.

El mantenimiento diario de la flota consiste en revisiones diarias relacionadas con las misiones de vuelo efectuadas por cada avión. Un avión debe ser sometido a una inspección luego de cada misión de vuelo efectuada. Además, períodicamente cada avión es sometido a mantenimientos preestablecidos. La ocurrencia de cada uno de estos mantenimientos dependerá de las horas acumuladas de vuelo para un determinado avión.

Cada avión es sometido a seis mantenimientos períodicos a los cuales se designa como Mantenimiento Tipo I, II,...,VI, mientras que labores de mantenimiento no planificadas

deben efectuarse en caso de fallas en un avión, las cuales se dividen en dos tipos diferentes según las necesidades de la reparación requerida. Cada mantenimiento tiene asignada una unidad operativa donde se lleva a cabo. La tabla 5.1 presenta la cantidad de horas después de las cuales se lleva a cabo cada mantenimiento y la estación donde ésta se realiza.

TABLA 5.1. Programa de mantenimiento según horas de vuelo

Actividad	Frecuencia	Unidad de Mantenimiento
Afinamientos menores	Después de cada misión	Escuadrón de combate
Mantenimento tipo I	Cada 50 horas de vuelo	Escuadrón de combate
Mantenimiento tipo II	Cada 125 horas de vuelo	Taller mecánico
Mantenimiento tipo III	Cada 250 horas de vuelo	Taller mecánico
Mantenimiento tipo IV	Cada 500 horas de vuelo	Almacén nacional
Mantenimiento tipo V	Cada 1.000 horas de vuelo	Almacén nacional
Mantenimiento tipo VI	Cada 2.000 horas de vuelo	Almacén nacional
Reparación de fallas I	Según necesidad	Escuadrón de combate
Reparación de fallas II	Según necesidad	Taller mecánico

La tabla 5.2 muestra las distribuciones de los tiempos requeridos para un avión en cada una de las actividades, según la cantidad de horas hombre requeridas, donde los T_i representan el tiempo medio de cada una de las distribuciones. Adicionalmente existe una cantidad máxima de mecánicos que puede ser asignada a un avión para cada una de las actividades.

Por lo tanto, para un avión en particular su operación consiste en lo siguiente: esperará en los hangares de su unidad primaria hasta que sea asignado a una misión, para lo cual son seleccionados según el tiempo de espera que llevan en los hangares. Posteriormente el avión lleva a cabo la misión correspondiente para luego regresar a la unidad donde se determina si debe recibir algún tipo de mantenimiento, ya sea rutinario o porque durante el vuelo presentó alguna falla (por simplicidad no se considera interrumpir una misión debido a la ocurrencia de alguna falla). En caso de requerir algún tipo de mantenimiento el avión

TABLA 5.2. Tiempos requeridos en cada actividad de mantenimiento

Actividad	Tiempo (HH)	Máximo de mecánicos
Mantenimiento tipo I	$Tria(T_1, 38, 68)$	4
Mantenimiento tipo II	$200 + Gamma(2, T_2)$	4
Mantenimiento tipo III	$500 + Gamma(2, T_3)$	4
Mantenimiento tipo IV	$1.300 + Gamma(2, T_4)$	5
Mantenimiento tipo V	$1.500 + Gamma(2, T_5)$	5
Mantenimiento tipo VI	$2.000 + Gamma(2, T_6)$	6
Reparación de fallas I	$Exp(T_7)$	3
Reparación de fallas II	$Exp(T_8)$	4

es trasladado a la unidad de mantenimiento correspondiente, donde espera hasta poder ser atendido, finalmente, se lleva a cabo el mantenimiento de rutina antes de regresar al hangar.

En cada unidad de mantenimiento existe un número fijo de mecánicos disponibles, mientras que un escuadrón de combate cuenta con 6 mecánicos, un taller mecánico contará con 15 y el almacén nacional tendrá 25 mecánicos.

Finalmente se realizaron los siguiente supuestos sobre las misiones de vuelo: los tiempos entre las distintas misiones tienen una distribución exponencial con media de 30 minutos, mientras que la duración de cada misión sigue una distribución normal con media 45 minutos y desviación estándar de 12. Además, cada misión requiere de un solo avión para realizarse.

5.2.2. Problema de Optimización

El objetivo del estudio de las políticas de mantenimiento en la flota finlandesa es poder optimizar este procedimiento, para lo cual, se define como valor a optimizar la cantidad media de aviones disponibles en los hangares, representada como D_i para cada hangar i, con $i = 1, 2, 3$. Para lo anterior, se realiza el supuesto de que se podrán mejorar los tiempos medios de atención en cada una de las unidades de mantenimiento a través de inversiones,

ya sea en capacitación adicional para los mecánicos, como en maquinarias y equipos que hagan más eficiente el trabajo.

Por lo anterior se define una función de utilidad donde el beneficio está representado por la cantidad de aviones media disponible, mientras que el costo será representado por la inversión necesaria para la mejora en cada uno de los tiempos.

El problema de optmización quedara definido de la siguiente manera:

$$Max\left\{b \cdot \overline{D} - c_i \cdot T_i - E\right\} \tag{5.1}$$

$$l_i \leq T_i \leq u_i \tag{5.2}$$

Donde b representa el beneficio marginal por cada unidad más de aviones disponibles, $\overline{D}$ el número medio de aviones disponibles en los hangares, cada c_i el costo marginal de disminuir en una unidad el tiempo medio de atención en cada proceso y E una constante que representa el costo fijo del problema.

A modo de ejemplo se estudió el sistema con el valor de las variables de decisión (cada T_i) en los valores máximos y mínimos posibles (es decir, en los l_i y los u_i respectivamente), los resultados se muestran en la tabla 5.3.

TABLA 5.3. Valor función de utilidad para configuraciones extremas

Valores T_i	$\overline{D}$	Fn. Objetivo
l_i	8,94	4.149,15
u_i	7,37	3.685,40

En ellos se puede ver la mejoría alcanzada al fijar todos los valores en el mínimo tiempo disponible, el cual reporta una mayor utilidad respecto al caso base (en que cada variable se fija en su valor mínimo, es decir, sin ninguna mejora en los tiempos) de $464,1$.

6. DESARROLLO DE EXPERIMENTOS

Para desarrollar los experimentos descritos fue necesario realizar la implementación, tanto de los algoritmos de optimización como de los sistemas de simulación, intentando que las condiciones fueran, dentro de lo posible, iguales para ambos mecanismos.

A continuación se describe el trabajo realizado.

Para el caso de la programación de las metodologías de optimización se aprovechó que se contaba con el algoritmo nHGA implementado a través del software Matlab[1]. Por lo tanto, buscando que la programación de cada uno de los algoritmos tuviera la menor incidencia posible en la comparación de desempeño, se programó el algoritmo ISS en el mismo software utilizando una notación similar.

Para validar la programación y el desempeño del algoritmo se realizaron pruebas con los mismos escenarios utilizados en Buchhloz y Thümmler (2005). Para esto se buscó optimizar la función esfera:

$$g(x) = -\left(\frac{x_1^2 + x_2^2}{8}\right) \tag{6.1}$$

a la cual se le agrega un ruido aleatorio con distribución Normal con media igual a cero y cuya varianza es diferente para distintos valores de x, quedando esta misma definida por la siguiente expresión:

$$r(x) = 0,2\left(1 + \frac{\sin(\pi x_1) + \sin(\pi x_2)}{4}\right) \tag{6.2}$$

Por lo tanto la función a optimizar será:

$$f(x) = g(x) + r(x) \cdot N(0,1) \tag{6.3}$$

con $x \in \Re^2$ en el espacio definido por $[-1,2]^2$.

Esta función tendrá su máximo en $x^* = (0,0)$ con $g(x^*) = 0$.

La figura 6.1 muestra los resultados obtenidos en el caso reción descrito comparando lo obtenido en el caso original versus los resultados obtenidos con la versión programada

[1]desarrollado por The MathWorks, Inc

en Matlab, considerando como medida de desempeño la función esfera original evaluada en los valores óptimos de las variables obtenidas en cada caso. Como se puede ver comportamiento de la nueva programación es muy similiar a la anterior.

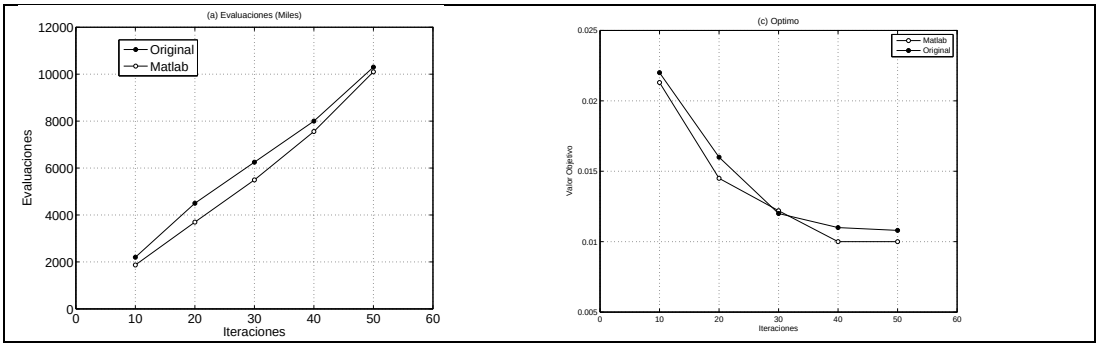

FIGURA 6.1. Validación de programación algoritmo ISS en matlab. a) Comparación de la medida de desempeño entre algoritmo ISS original y versión programada en Matlab. b) Comparación de la cantidad de evaluaciones entre algoritmo ISS original y versión programada en Matlab.

Las diferencias se deben especialmente a la cantidad de muestras obtenidas en este caso, que es inferior a las del trabajo original por lo que los valores obtenidos presentan un mayor margen de error. Adicionalmente la figura 6.1 muestra también la similitud en cantidad de evaluaciones del sistema de simulación en ambos casos.

Con lo anterior se valida la programación del algoritmo ISS.

Un factor relevante a la hora de comparar el desempeño de las metodologías de optimización es la plataforma en la cual se implementan los modelos de simulación ya que la eficiencia en la solución dependerá fuertemente de en qué programa y cómo se realiza la evaluación de la simulación (Pasupathy & Henderson, 2006).

Para la modelación de los sistemas a simular se optó por utilizar en todos los casos el software de simulación Arena[2], debido a que es un programa desarrollado específicamente para la simulación y presenta las herramientas necesarias para el correcto tratamiento de los problemas a resolver. Adicionalmente como OptQuest utiliza Arena se busca que todas las simulaciones se realicen en el mismo ambiente. Por lo anterior se realizó la conexión de

[2]Software desarrollado por Rockwell Automation

Matlab con Arena de tal modo que el primero pudiera controlar el modelo de simulación y obtener la información relevante en cada simulación.

Finalmente es importante destacar que todos los procedimientos anteriormente descritos fueron realizados por un usuario sin conocimientos avanzados en programación, lo que implica que, en caso de ser programados por un experto, podrían mejorar algunos procesos y el tiempo requerido por cada algoritmo.

Como se podrá observar en el capítulo de resultados, el tiempo requerido para resolver cada uno de los problemas con los algoritmos es muy elevado, especialmente en el caso del algoritmo genético nHGA, por lo cual el tamaño de la muestra utilizada para compararlos debió ser inferior a lo esperado, teniendo que utilizarse, para cada configuración diferente, un tamaño de muestra de 10. Es importante notar que, pese a lo bajo de este valor se requirió de más de tres meses para obtener estos resultados.

Lamentablemente, esta limitación originada por los altos tiempos de solución, no permitió obtener una mayor base de resultados para comparar las diferentes metodologías. Lo cual, adicionalmente, restringió los estudios que fue posible realizar, impidiéndo analizar en profundidad la incidencia de los diferentes parámetros de cada uno de los algorítmos en su desempeño.

7. RESULTADOS

Para evaluar el funcionamiento de cada uno de los algoritmos en los problemas anteriormente descritos se definirán las siguientes medidas de desempeño:

- En primer lugar la cantidad de evaluaciones del sistema de simulación utilizadas en la resolución.
- En segundo lugar el tiempo utilizado en resolver el problema, el cual estará directamente relacionado con la cantidad de evaluaciones.
- Finalmente la calidad del resultado obtenido con cada metodología; Como en estos casos no se conoce el resultado óptimo de cada uno de los problemas, solamente se podrá observar cual de los resultados entrega un mayor valor para la función objetivo y las relaciones entre cada uno de estos resultados, pero no su distancia al óptimo real.

Para que los resultados fueran comparables, todas las pruebas se corrieron en un Servidor PowerEdge 2950 con dos procesadores Intel Xeon Quadcore E5320 de 1,86 GHz, 8GB de memoria Ram, disco duro de 1,5 TB en RAID 5 y Sistema operativo Microsoft Windows Server 2003 R2 Enterprise x64 Edition, Service Pack 2.

7.1. Sistema 1: Abastecimiento de Combustible Ejército Turco

Bajo las condiciones anteriormente descritas se utilizaron los dos algoritmos para maximizar el tiempo durante el cual el sistema puede satisfacer la demanda. Los parámetros fueron fijados del modo más similar posible en ambos casos, para que así que sus resultados sean comparables. Como ejemplo se definió la cantidad inicial de evaluaciones para cada individuo $n_0 = 20$ para ambos casos.

Para el caso de la estrategia evolutiva, ISS, se fijaron los siguiente parámetros; tamaño de la población $N = 20$, cantidad inicial de evaluaciones para cada individuo $n_0 = 20$, cantidad de crías en cada nueva iteración $\lambda = 5$, probabilidad de selección correcta para ranking y selección $P^* = 0,9$, parámetro de indiferencia $\delta = 0,1$ y tamaño de la población

de elite $\omega = 3$. Adicionalmente se tiene un parámetro controlable para la heurística, según la cantidad de iteraciones hasta detener el algoritmo. Se estudiaron, en primer lugar, los resultados variando dicho parámetro.

Los resultados de utilizar esta estrategia evolutiva se pueden observar en la figura 7.1. En ella se puede ver el aumento progresivo del número de evaluaciones a medida que aumentan las iteraciones. Lo mismo ocurre con el tiempo necesario para obtener una solución, ya que esta variable está directamente relacionada con la cantidad de evaluaciones del sistema de simulación. Se puede ver que, en el caso del aumento a 70 iteraciones, el número adicional de evaluaciones es inferior a los incrementos anteriores. Lo anterior se explica ya que las nuevas iteraciones no logran aportar individuos que mejoren el desempeño y que requieran mayor número de iteraciones del procedimiento de ranking y selección.

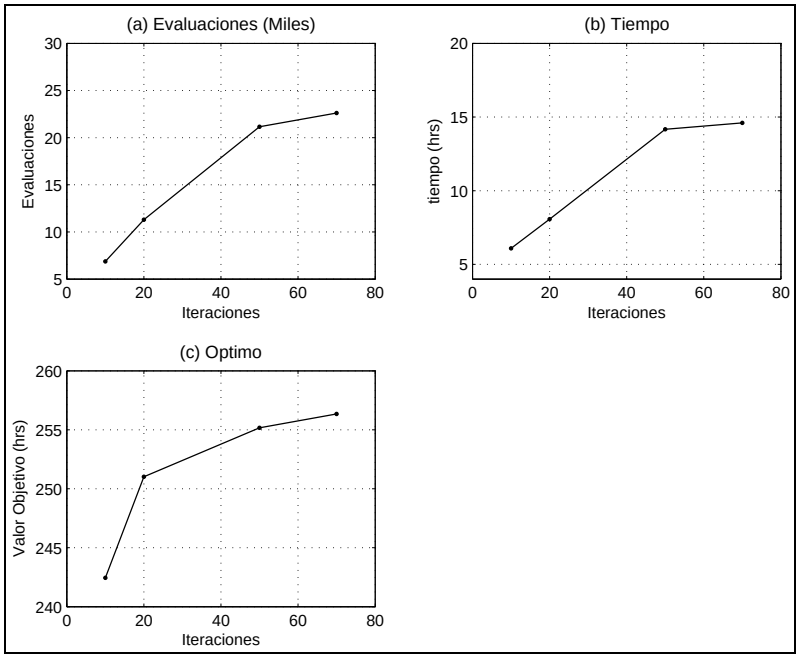

FIGURA 7.1. Desempeño algoritmo ISS en problema de abastecimiento de combustible: (a) Evaluaciones promedio según número de iteraciones, (b) Tiempo promedio de ejecución según número de iteraciones, (c) Valor óptimo promedio según número de iteraciones.

Lo anterior se ve reflejado en la mejora de la función objetivo, que es cada vez menor, según lo cual se identifica que sobre 60 iteraciones se logran mejoras poco significativas en

la función objetivo. Esto refleja cómo, a medida que aumenta el número de iteraciones, el crecimiento marginal en el valor de la función objetivo se vuelve decreciente. Lo cual es consistente con las heurísticas de optimización, según las cuales, en una primera instancia se logrará un avance rápido hacia el óptimo, pero haciendo que cada vez cueste más superar el mejor resultado obtenido hasta el momento.

La figura 7.2 muestra la evolución del mejor valor encontrado hasta el momento (individuo mejor adaptado) en función de la cantidad de iteraciones realizadas. En ella se puede observar que el mejor valor varía en pocas ocaciones a partir de la iteración número 50, reflejando la dificultad de encontrar individuos mejor adaptados. La alta variabilidad de los resultados durante las primeras iteraciones se explican ya que a medida que aumentan estas mismas, se va obteniendo un valor cada vez más cercano a la media real para un determinado individuo, explicando así que el valor de la función objetivo pueda cambiar aún cuando el mejor individuo encontrado hasta el momento no haya variado de una iteración a la otra.

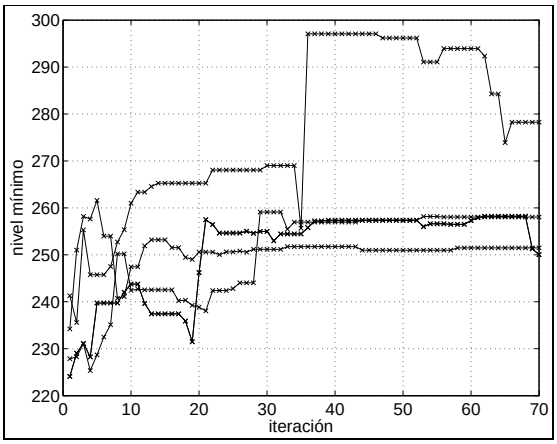

FIGURA 7.2. evolución del mejor resultado obtenido de acuerdo al número de iteraciones

Para el caso del algorítmo genético nHGA se intentó analizar su desempeño con los parámetros tal como se proponen en el trabajo de Olguin (2008), sin embargo, se pudo concluir que bajo esta configuración no se podía obtener resultados en un tiempo satisfactorio, de hecho, luego de más de 36 horas en ejecución aún no se podía obtener alguna estimación. Sobre lo anterior se deben tener las siguientes consideraciones:

- Se propone utilizar un tamaño de población $N = 15n$, con n la cantidad de variables del problema. En este caso $n = 20$ y por lo tanto $N = 300$.
- La cantidad máxima de iteraciones permitidas para cada etapa se propone fijarla en $50n^2$, correspondiente a 20.000 iteraciones en la etapa de diversificación y otras 20.000 en la de intensificación.

Esto demuestra que, el tener ligados ciertos parámetros del algoritmo a las variables de decisión con el fin de lograr un equilibrio en la cobertura del espacio solución o un número adecuado de iteraciones, puede reflejar buenos resultados cuando la dimensión del problema es pequeño, sin embargo, a medida que ésta aumenta, los parámetros crecerán excesivamente, dificultando la resolución y restando eficiencia al algoritmo.

Para poder hacer abordable el sistema a través de este algoritmo, se debieron modificiar estos valores, para lo cual se definió el tamaño de la población $N = 40$ y la cantidad máxima de iteraciones en cada etapa como $2Nn$, es decir 1.600 iteraciones para cada etapa. Además, los criterios para ranking y selección se fijaron en $P^* = 0,9$ y $\delta = 0,1$, así los parámetros de ranking y selección son iguales a los utilizados con el algoritmo anterior.

Para este caso los resultados de los experimentos son los siguientes: el tiempo promedio utilizado para resolver el problema es de $21,25$ horas, mientras que, en promedio se necesitaron $49.148,5$ evaluaciones del sistema de simulación. Sin embargo, el valor de la función objetivo mejora considerablemente obteniéndose un valor de $268,4$, un $4,2\%$ mejor que el obtenido con ISS.

Lamentablemente, el problema de simulación no pudo ser resuelto utilizando OptQuest para Arena, debido a limitaciones técnicas del software que no permiten utilizar los niveles mínimos de cada tanque como variable de decisión.

En la tabla 7.1 se pueden observar los resultados obtenidos con cada una de las metodologías probadas. En ellos se puede ver que claramente el desempeño del algoritmo genético nHGA requiere de mayores recursos computacionales (mayor cantidad de evaluaciones y por lo tanto mayor tiempo medio de ejecución) pero a su vez entrega mejores resultados.

TABLA 7.1. Medidas de desempeño para cada algoritmos utilizado

	Cantidad de iteraciones	Evaluaciones	Tiempo (hrs)	Valor función objetivo
ISS	10	6.872	6,08	242,25
ISS	20	11.299	8,06	251,02
ISS	50	21.160	14,16	255,17
ISS	70	22.614	14,60	256,66
nHGA	n/a	49.149	21,25	268,41

Es importante notar que, para el caso de la estrategia evolutiva ISS, el mejor valor encontrado para la función objetivo supera tan solo en un $0,6\%$ el mejor valor encontrado al fijar los niveles mínimos de todos los tanques en el mismo valor (particularmente en un 25%) como se pudo observar en la figura 5.2. Mientras que el algoritmo genético nHGA logra una mejora de un $5,25\%$ respecto a dicho valor.

Debido a la gran cantidad de simulaciones durante cada optimización y a que es necesario pasar de un software a otro para cada una de estas simulaciones, se estudió el tiempo requerido en cada simulación desde el momento en que el algoritmo decide obtener una nueva evaluación del sistema con cierta configuración. Se obtuvo que el tiempo medio de cada evaluación es de $1,99$ segundos. Este tiempo puede ser mejorado cambiando la programación, como por ejemplo podría implementarse el problema directamente en el mismo programa que los algoritmos, lo cual permitiría un ahorro considerable en tiempo. Sin embargo, la comparación es válida ya que para ambos algoritmos se utilizó la misma metodología, el mismo programa y el mismo sistema de simulación.

La tabla 7.2 muestra el mejor resultado obtenido con cada uno de los algoritmos.

A pesar de las modificaciones impuestas para los parámetros de nHGA, se puede observar su superioridad respecto a ISS al entregar en promedio mejores resultados que este último. En cuanto al mejor resultado obtenido por cada una de las metodologías, se puede ver que éstas son muy similares entre ambas, sin embargo ISS utilizó un 62% menos de

TABLA 7.2. Mejor resultado obtenido con cada metodología

	Tiempo (hrs)	Evaluaciones	Valor función objetivo	Cantidad de iteraciones
ISS	21,12	34.645	278,72	70
nHGA	40,78	91.763	279,88	n/a

evaluaciones del sistema de simulación que las requeridas por nHGA y tan solo un 48% del tiempo.

El principal problema en la optimización del sistema descrito radica en la medida de desempeño utilizada para la función objetivo, debido a que ésta corresponde al largo de la simulación, que representa el tiempo durante el cual el sistema es capaz de satisfacer toda la demanda. Por cada evaluación del sistema se obtiene un solo valor para esta medida, el cual presenta una alta variabilidad y para el cual el supuesto de las metodologías de ranking y selección, de que el valor objetivo corresponde a una distribución normal, según el teorema de los grandes números, no es aplicable. Sin embargo, se decidió utilizar igualmente los algoritmos como si los supuestos se cumplieran, ya que representa una aplicación real de un problema de simulación que no puede ser adaptado para que se cumplan las condiciones. Debido a lo anterior, la varianza de las observaciones es altamente significativa y debe ser considerada en caso de querer aplicar los resultados obtenidos.

Por dicha razón, se definió el número de evaluaciones iniciales n_0 en 20 en ambos casos, para así disminuir la varianza de las observaciones y disminuir el intervalo de confianza para cada uno de los individuos de la población.

Merece una mención especial el tiempo requerido en cada caso para realizar la optimización. En los casos de mejor desempeño se requieren en promedio 21 y 18 horas en cada uno de los casos, lo que impidió poder realizar una mayor cantidad de muestras para las comparaciones.

7.2. Sistema 2: Mantención de Aviones de la Flota Finlandesa

Para este caso en particular se tiene que la salida del sistema de simulación, de la cual depende la función objetivo, será la cantidad media de aviones disponibles en cada uno de los hangares durante el tiempo de simulación, por lo tanto, por el teorema de los grandes números, se puede suponer que esta variable presenta una distribución normal y, por consiguiente, los supuestos de ranking y selección se pueden asumir verdaderos. Por lo anterior se define el número inicial de evaluaciones para cada individuo en $n_0 = 5$.

Para utilizar la estrategia evolutiva, ISS, se fijaron los siguientes parámetros: Tamaño de la población $N = 20$, cantidad mínima de evaluaciones para cada individuo $n_0 = 5$, crías por cada iteración $\lambda = 5$, probablidad de selección correcta $P^* = 0,9$, parámetro de indiferencia $\delta = 0,1$ y tamaño población de elite $\omega = 3$.

Nuevamente se estudió el desempeño del algoritmo a medida que se aumentaba la cantidad de iteraciones del mismo. Los resultados se pueden obervar en la figura 7.3, donde se observa que, como era de esperar, el valor de la función objetivo mejora a medida que aumenta la cantidad de iteraciones.

En el caso de este problema, al utilizar el algoritmo genético nHGA se realizó con la configuración propuesta originalmente (Olguin, 2008), lo cual fue posible debido a que la cantidad de variables es menor a la del problema anterior, ya que en este caso el tamaño de la población es de $N = 105$ y la cantidad máxima de iteraciones por etapa de 2.450.

Con esta configuración se obtuvieron los siguientes resultados: la cantidad promedio de evaluaciones es de 37.491, mientras que el tiempo medio requerido alcanzó las 23,28 horas, con esto el valor medio de la función objetivo es de 4.210.

A continuación se estudió el desempeño del software de optimización OptQuest, en su aplicación desarrollada especialmente para el software Arena. Tal como se explicó en la sección 4.3 fue necesario definir un criterio de precisión, según lo cual se estableció que se realizaría un número variable de simulaciones por cada iteración entre 5 y 20. Como

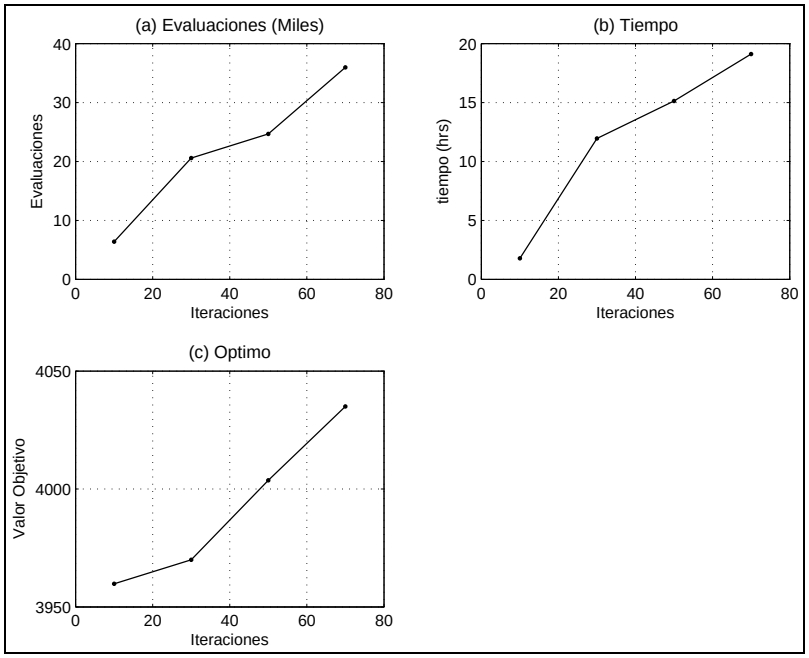

FIGURA 7.3. Medidas de desempeño para ISS en problema de flota aérea: (a) Evaluaciones promedio según número de iteraciones, (b) Tiempo promedio de ejecución según número de iteraciones, (c) Valor óptimo promedio según número de iteraciones.

criterio de parada se estipuló una cantidad máxima de iteraciones, la cual se varió para observar el desempeño de esta metodología.

La figura 7.4 muestra los resultados obtenidos utilizando OptQuest. Debido a los menores tiempos requeridos por este programa, se pudo variar ampliamente la cantidad máxima de iteraciones, pudiendo observarse un aumento de 188 en la utilidad al pasar de 100 a 300 iteraciones, mientras que esta diferencia disminuye considerablemente a medida que se aumentan las iteraciones, a modo de ejemplo, al pasar de 1.000 a 3.000 iteraciones la utilidad media solo mejora 13 unidades. Este hecho muestra nuevamente cómo la utilidad marginal de una nueva iteración es decreciente a medida que el número de iteraciones aumenta. La tabla 7.3 muestra los resultados medios obtenidos con cada una de las metodologías. Para el caso de OptQuest el tiempo es un valor estimado ya que no fue posible obtener su valor exacto como en el caso de las otras metodologías, mientras que la cantidad de evaluaciones, también para OptQuest representan el máximo posible de evaluaciones que podría utilizar el programa.

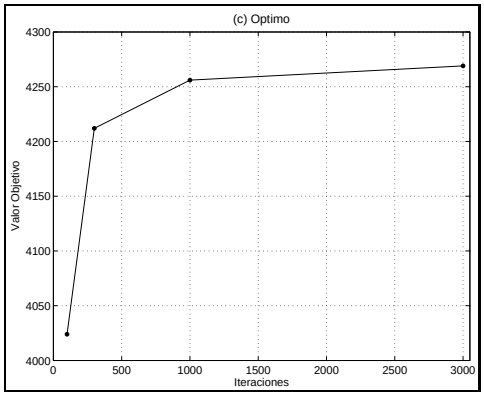

FIGURA 7.4. Valor óptimo promedio según número de iteraciones utilizando OptQuest para Arena.

TABLA 7.3. Resultado promedio obtenido con cada metología para problema de flota aérea finlandesa.

	Cantidad de Iteraciones	Tiempo* (hrs)	Evaluaciones**	Valor función objetivo
ISS	10	3,8	6.400	3.960
ISS	30	11,95	20.576	3.970
ISS	50	15,13	24.667	4.004
ISS	70	19,11	35.968	4.035
nHGA	n/a	23,28	37.895	4.210
OptQuest	100	0,24	1.250	4.024
OptQuest	300	0,72	3.750	4.212
OptQuest	1.000	2,39	12.500	4.256
OptQuest	3.000	7,17	37.500	4.269

* Para OptQuest valores estimados.

* * Para OptQuest máximo posible.

Se puede observar que los mejores resultados se obtienen utilizando OptQuest, el cual, adicionalmente presenta tiempos de ejecución muy inferiores a los de las otras metodologías. Por ejemplo, comparando el tiempo del algoritmo genético nHGA con el desempeño de OptQuest con 300 iteraciones, el valor medio de la función objetivo presenta sólo un $0,05\%$

de diferencia, mientras que el tiempo utilizado por OptQuest es un 97% inferior al requerido por nHGA.

En cuanto a las diferencias en el tiempo, es necesario tener en cuenta que OptQuest es una estrategia desarrollada especialmente para Arena y que, por lo tanto, manipula y se conecta con este último de manera más eficiente que la utilizada por las otras dos metodologías, sin embargo, también se debe considerar que OptQuest utiliza menos evaluaciones de la función objetivo para obtener un mejor resultado.

TABLA 7.4. Mejor resultado con cada metología para problema de flota aérea finlandesa.

	Cantidad de Iteraciones	Tiempo (hrs)	Evaluaciones	Valor función objetivo
ISS	70	20,37	35.827	4.166
nHGA	n/a	22,50	36.630	4.333
OptQuest	3.000	n/d	n/d	4.321

Finalmente la tabla 7.4 muestra el mejor resultado obtenido con cada una de las metodologías. Donde se puede ver que el mejor valor para la función objetivo se obtuvo utilizando el algoritmo genetico nHGA, el cual es un 4% superior al mejor resultado obtenido con la estrategia evolutiva ISS, y un $0,2\%$ superior al mejor resultado obtenido con OptQuest. Como ya se mencionó para esta tabla los valores para el tiempo y el número de evaluaciones utilizando OptQuest no está disponible.

8. CONCLUSIONES

Este trabajo presenta el desempeño de metaheurísticas para optimización via simulación desarrolladas en los últimos años al intentar resolver problemas reales, buscando así entregar mayor información sobre la eficiencia de los mismos, para entregar a los usuarios interesados directrices sobre cuál de ellos utilizar en caso de enfrentarse a algún problema de optimización vía simulación. En los resultados se puede observar el aumento considerable en el tiempo requerido por los algoritmos para solucionar los diferentes problemas a medida que aumenta la cantidad de variables a definir, mostrando así que podría ser equivocado comparar algoritmos en pequeños escenarios de prueba sin estudiar su desempeño en sistemas más complejos.

A modo de resumen entre los diferentes algoritmos probados se pueden destacar los siguientes puntos:

- Al comparar el valor de la función objetivo el algoritmo nHGA muestra un mejor desempeño que el obtenido utilizando ISS (con 70 iteraciones), mostrando mejoras de un $4,6\%$ y un $4,3\%$ respectivamente.
- Como contraparte a lo anterior, cantidad de evaluaciones del sistema de simulación es superior con nHGA requiriendo, al utilizar ISS (con 70 iteraciones), un 46% menos de evaluaciones para el caso del problema del ejército turco y un 5% menos de evaluaciones en el problema de la flota aérea finlandesa.
- Se reconoce la incidencia en el criterio de término para ISS en su desempeño, lamentablemente no fue posible profundizar más en la mejora que se podría obtener al seguir aumentando la cantidad de iteraciones.
- El software de optimización OptQuest para Arena demuestra tener mayor eficiencia, para los casos en que puede efectivamente resolver el problema, entregando una mejora de 234 unidades en la utilidad promedio obtenida utilizando ISS con 70 iteraciones y de 59 unidades al compararse con nHGA para el problema de la flota aérea finlandesa.

El algoritmo genético nHGA presenta la dificultad de que, a medida que aumentan las variables, el tamaño de la población y la cantidad de iteraciones en las etapas de intensificación y diversificación aumentan considerablemente haciendo prácticamente imposible utilizarlo bajo las características iniciales. Sin embargo, después de algunas modificaciones a estas variables se observa que se validan los resultados mostrados en Olguin (2008) según los cuales el desempeño de este algoritmo supera a los presentados por la estrategia evolutiva ISS desarrollada en Buchhloz y Thümmler (2005).

Los mejores resultados se obtuvieron utilizando el software de simulación OptQuest para Arena, el cual incluso con un reducido número de iteraciones y en tiempos inferiores al 40% de los utilizados por nHGA y por ISS entrega en promedio los mejores resultados.

A pesar de lo anterior OptQuest no es capaz de solucionar cualquier tipo de problemas, a modo de ejemplo no fue posible utilizarlo para el problema de abastecimiento del Ejército Turco, debido a limitaciones en cuanto a los valores que puede utilizar como variables de decisión. En este sentido los dos algoritmos evolutivos utilizados presentan una gran flexibilidad, debido a que cualquier variable del sistema de simulación puede ser utilizado como variable de decisión.

Una mención especial requieren las limitaciones en cuanto a la gran cantidad de recursos computacionales necesarios para resolver cada uno de los problemas. Ambas estrategias demostraron entregar mejoras en los sistemas a optimizar, sin embargo no será eficientes utilizarlas en caso de requerirse una toma de decisiones constante y en corto tiempo, como lo plantea la simulación *on-line*.

Estas mismas limitaciones hicieron imposible probar la incidencia de diferentes parámetros de los algoritmos en la calidad de sus soluciones, por ejemplo el tamaño de la población de elite (ω) en la estrategia ISS o el tamaño de la población y la cantidad de evaluaciones iniciales en ambos casos, así como también los diferentes parámetros utilizados para ranking y selección.

En este aspecto, una tarea pendiente es la de comparar el desempeño de ambos algoritmos bajo condiciones en que ambos utilicen la misma cantidad de determinado recurso,

por ejemplo, cuando ambos requieran las misma cantidad de evaluaciones del sistema de simulación o una cantidad similar de crías generadas durante su solución. Durante el desarrollo de este trabajo, se buscó que las condiciones de comparación fueran lo más similares posibles en ambos casos, pero queda planteada la inquietud de cómo sería la comparación entre el valor de la función objetivo al equipararse las condiciones de ambos algoritmos según lo aquí indicado.

El trabajo desarrollado demuestra que algoritmos que pueden parecer altamente eficientes en problemas de laboratorio pueden disminuir su eficiencia al momento de resolver problemas de mayor complejidad. Especialmente en cuanto al tiempo y a la cantidad de evaluaciones del sistema de simulación utilizado. Esto se ve más claramente en el caso de los parámetros dependientes de la dimensión del espacio de solución para nHGA, los cuales, al crecer junto con esta dimensión, llevan a que no se pueda resolver el problema, incluso en un tiempo de más de 48 horas.

En la elección de los problemas a resolver se buscó que fueran diferentes entre ellos para así lograr mayor variedad en los resultados obtenidos tal como se propone en el trabajo de Whitley et al. (1996) que postula que al probar algortimos de búsqueda los resultados dependerán tanto de los algoritmos como de los problemas a resolver, por lo tanto estos últimos deben ser desafiantes y diversos.

Adicionalmente se postula para una investigación futura combinar características de ambos algoritmos para obtener mejores resultados. Una importante mejora que se plantea introducir al algoritmo ISS es cambiar la metodología de selección de los padres, pasando del método de selección aleatoria por uno que asigne mayor probabilidad de ser elegido a aquellos individuos mejor adaptados.

Algo no mencionado con anterioridad es que, en la variabilidad de los resultados obtenidos con cada metodología, se pudo observar claramente la dependencia de los resultados de la población inicial en cada uno de los casos, especialmente para la estrategia evolutiva ISS. Para paliar esta deficiencia se propone incorporar en esta estrategia algún mecanismo que asegure una cobertura lo más homogénea posible del espacio de solución

por parte de la población inicial, para lo cual se podría recoger la propuesta de una vecindad como la utilizada en nHGA.

Una idea que ha ido tomando fuerza en las investigaciónes y que podría mejorar los tiempos requeridos por ambos algoritmos aquí utilizados en la del análisis de factores, *factor screening*, consistente en identificar qué variables de decisión tienen un mayor efecto en las variables de salida relevantes para así disminuir la dimensionalidad del problema. Así, se podrían llevar problemas de mayores dimensiones a dimensiones significativamente menores que permitan una resolución más rápida y eficiente.

Otro factor cuyo efecto podría ayudar a reducir la cantidad de evaluaciones es el de modificar los algoritmos de modo de fijar los recursos computacionales disponibles para cada iteración, de modo que, aún cuando no haya logrado terminar completamente los procesos secuenciales de ranking y selección, pasen a la siguiente iteración evitando un aumento desproporcionado del número de evaluaciones respecto a la mejora obtenida.

Entre las dificultades que se debió enfrentar para el desarrollo de los experimentos, se debió solucionar el problema propuesto por Azadivar (1999) según el cual comunicar softwares de simulación con rutinas de optimización puede ser una tarea difícil. El tiempo necesario para cada evaluación, y por lo tanto el desempeño de un determinado algoritmo, dependerá fuertemente de la calidad de esta comunicación. Debido a las deficiencias que podría presentar el trabajo realizado en este aspecto es que se indica tener precaución al observar los resultados, en cuanto a tiempo, respecto a los obtenidos con OptQuest, ya que éstos estarán fuertemente influenciados por esta programación que podría ser mejorada.

Finalmente es importante destacar el avance alcanzado en esta investigación al enfrentar problemas de mayor dimensionalidad con los algoritmos desarrollados en los últimos años. Para quien desee aplicar la optimización vía simulación por medio de estos algoritmos podría verse en la dificultad que su problema tiene más variables que las de los aquí presentados, ya que, de hecho, su dimensionalidad es baja respecto a problemas comunes de optimización determinística e incluso estocástica. Por lo que las dificultades enfrentadas

especialmente en cuanto al tiempo de ejecución, podrían entregar lineamientos a un usuario interesado.

BIBLIOGRAFIA

Andradottir, S. (1998a). A review of simulation optimization techniques. *Proceedings of the 1998 Winter Simulation Conference*, 151–158.

Andradottir, S. (1998b). Simulation Optimization. En J. Banks (Ed.), *Handbook of simulations: Principles, methodology, advances, applications, and practice* (pp. 307–333). John Wiley.

April, J., Better, M., Glover, F., Kelly, J. & Laguna, M. (2006). Enhancing business process management with simulation optimization. *Proceedings of the 2006 Winter Simulation Conference*, 642–649.

April, J., Glover, F., Kelly, J. & Laguna, M. (2003). Practical introduction to simulation optimization. *Proceedings of the 2003 Winter Simulation Conference*(MM-NN).

Azadivar, F. (1999). Simulation optimization methodologies. *Proceedings of the 1999 Winter Simulation Conference*, 93–100.

Bäck, T. (1996). *Evolutionary algorithms in theory and practice*. U.S.A.: Oxford University Press.

Ballester, P., & Carter, J. (2003). Real-parameter genetic algorithms for finding multiple optimal solutions in multi-modal optimization. *Genetic and Evolutionary Computation Conference, Lecture Notes in Computer Science*, 706–717.

Ballester, P., & Carter, J. (2004). An effective real-parameter genetic algorithm with parent centric normal crossover for multimodal optimization. *Proceeding of the Genetic and Evolutionary Computation Conference*, 901–913.

Bechhofer, R. (1954). A single-sample multiple decision procedure for ranking means of normal populations with known variances. *The Annals of Mathematical Statistics*, *25*(1), 16–39.

Bechhofer, R., Santner, T. & Goldsman, D. (1995). *Design and analysis for statistical selection, screening and multiple comparison*. New York: John Wiley and Sons.

Bessaou, M., & Siarry, P. (2001). A genetic algorithm with real-value coding to optimize multimodal continuous functions. *Structural and Multidisciplinary Optimization*, *23*(1), 63–74.

Beyer, H., & Schwefel, H. (2002). Evolution strategies a comprehensive introduction. *Natural Computing*, *1*, 3–52.

Boesel, J., Nelson, B. & Kim, S. (2003). Using ranking and selection to ”clean up” after simulation optimization. *Operations Research*, *51*(5), 814–825.

Box, G., & Wilson, K. (1951). On the experimental attainment of optimum conditions. *Journal of the Royal Statistical Society. Series B (Methodological)*, *13*(n. 1), 1–45.

Branke, J., Chick, S. & Schmidt, C. (2005). New developments in ranking and selection: An empirical comparison of the three main approaches. *Proceedings of the 2005 Winter Simulation Conference*, 708–717.

Buchhloz, P., & Thümmler, A. (2005). Enhancing evolutionary algorithms with statistical selection procedures for simulation optimization. *Proceedings of the 2005 winter simulation Conference*, 842–852.

Carson, Y., & Maria, A. (1997). Simulation optimization: Methods and applications. *Proceedings of the 1997 Winter Simulation Conference*, 118–126.

Chelouah, R., & Siarry, P. (2000). A continuous genetic algorithm designed for the global optimization of multimodal functions. *Journal of Heuristics*, 191–213.

Chen, E., & Kelton, W. (2000). An enhanced two-stage selection procedure. *Proceedings of the 2000 Winter Simulation Conference*, 727–735.

Chen, H., Chen, C., Lin, J. & Yücesan, E. (1999). An asymptotic allocation for simultaneous simulation experiments. *Proceedings of the 1999 Winter Simulation Conference*, 359–366.

Deb, K., & Agrawal, R. (1995). Simulated binary crossover for continuous search space. *Complex Systems*, *9*(2), 115–148.

Dudewicz, E., & Dalal, S. (1975). Allocation of observations in ranking and selection with unequal variances. *The Indian Journal of Statistics*, *37*(1), 28–78.

Dummler, M. (1999). Using simulation and genetic algorithms to improve cluster tool performance. *Proceedings of the 1999 Winter Simulation Conference*, 875–879.

Eglese, W. (1990). Simulated annealing: A tool for operational research. *European Journal of Operational Research, 46*, 271–281.

Eshelman, L., Caruna, R. & Schaffer, J. (1989). Biases in the crossover landscape. En J. Schaffer (Ed.), *Proceedings of th 3rd international conference on genetic algorithms and their applications* (pp. 10–19). Morgan Kaufmann Publishers.

Fu, M. (2002). Optimization for simulation: Theory vs. practice. *INFORMS Journal on Computing, 14*(n. 3), 192–215.

Fu, M., Adradóttir, S., J.S.Carson, Glover, F., Harrel, C., Ho, Y. et al. (2000). Integrating optimization and simulation: Research and practice. *Proceedings of the 2000 Winter Simulation Conference*, 610–616.

Fu, M., April, J. & Glover, F. (2005). Simulation optimization: A review, new developments and applications. *Proceedings of the 2005 Winter Simulation Conference*, 83–95.

Fu, M., & Hu, J. (1996). A comparison of perturbation analysis techniques. *Proceedings of the 1996 Winter Simulation Conference*, 295–301.

García-Martínez, C., Lozano, M., Herrera, F., Molina, D. & Sánchez, A. (2005). *Chromosome differentiation for the application of parent-centric real-parameter crossover operators* (Tech. Rep.). University of Granada.

Gazmuri, P., & Olguín, J. (2007). *Blending a genome differentiated, real-coded genetic algorithm with a nelder-mead simplex search algorithm for the efficient optimization of continuous functions.* (Working Paper, Escuela de Ingeniería Pontificia Universidad Católica de Chile)

Glasserman, P. (1991). *Gradient estimation via perturbation analysis*. Boston: Springer.

Glynn, P. (1990). Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM, 33*(10), 75–84.

Goldberg, D., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. En G. J.E. & Rawlins (Eds.), *Foundations of genetic algorithms* (pp. 307–333). Morgan Kaufmann Publishers.

Goldsman, D., & Nelson, B. (1998). Comparing Systems via Simulation. En J. Banks (Ed.), *The handbook of simulation* (pp. 307–333). John Wiley.

Grewal, C., Rogers, P. & Enns, S. (2008). Simulation optimization of supply chain replenishment strategies using OptQuest. *Proceedings of Sixth International Symposium on Supply Chain Management*, 214–229.

Gurkan, G., Yonca, A. & Robinson, S. (1994). Sample-path optimization in simulation. *Proceedings of the 1994 Winter Simulation Conference*, 247–254.

Hartmann, M. (1991). An improvement on Paulson´ procedure for selecting the population with the largest mean from k normal populations with a common unknown variance. *Sequential Analysis*, *10*, 1–16.

Ho, Y., Sreenivas, R. & Vakili, P. (1992). Ordinal optimization of deds. *J. Discrete Event Dynamic Systems*, *2*(2), 61–88.

Holland, J. (1975). *Adaptation in natural and artificial systems*. U.S.A.: The University of Michigan Press.

Hood, S., & Welch, P. (1993). Response surface methodology and its application in simulation. *Proceedings of the 1993 Winter Simulation Conference*, 115–122.

Hu, N. (1992). Tabu search method with random moves for globally optimal design. *International Journal for Numerical Methods in Engineering*, *35*, 1055–1070.

Hwang, S., & He, R. (2006). A hybrid real-parameter genetic algorithm for function optimizaton. *Advanced Engineering Informatics*, *20*, 7–21.

Kabirian, A., & Olafsson, S. (2007). Allocation of simulation runs for simulation optimization. *Proceedings of the 2007 Winter Simulation Conference*, 363–371.

Kim, S. (2007). Recent advances in ranking and selection. *Proceedings of the 2007 Winter simulation Conference*, 162–172.

Kim, S., & Nelson, B. (2001). A fully sequential procedure for indifference-zone selection in simulation. *ACM Trnasactions on Modeling and Computer Simulation, 11*(3), 251–273.

Kleijnen, J. (2008). Response surface methodology for constrained simulation optimization: An overview. *Simulation Modelling Practice and Theory, 16,* 50–64.

Kleijnen, J., & Wan, J. (2007). Optimization of simulated systems: OptQuest and alternatives. *Simulation Modelling Practice and Theory*(n. 15), 354–362.

Kushner, H., & Yin, G. (1997). *Stochastic aproximation algorithms and applications*. New York, U.S.A.: Springer-Verlag.

Laguna, M. (1997). Metaheuristic optimization with evolver, genocop and optquest. *EURO/INFORMS Joint International Meeting, Plenaries and Tutorials*, 141–150.

Laguna, M., & Marti, R. (2002). Optimization software class libraries. En S. Voss & D. Woodruff (Eds.), *Optimization software class libraries.* Kluwer Academic Publishers.

Law, A. M., & Kelton, W. (2000). *Simulation modeling and analysis* (3^{rd} ed.). U.S.A.: Mc Graw Hill Higher Education.

Lee, S., Khoo, L. & Yin, X. (2000). Optimising an assembly line through simulation augmented by genetic algorithms. *The International Journal of Advanced Manufacturing Technology*, 220–228.

Müller, S., Sbalzarini, I., Walther, J. & Koumoutsakos, P. (2001). Evolution strategies for the optimization of microdevices. *Proceedings of the Congress on Evolutionary Computation*, 302–309.

Nelder, J., & Mead, R. (1965). A simplex method for fuction minimization. *Computer Journal, 7*(4), 1044–1051.

Nelson, B., Swann, J., Goldsman, D. & Song, W. (2001). Simple procedures for selecting the best simulated system when the number of alternatives is large. *Operations Research, 49,* 950–963.

Olguin, J. (2008). *Algoritmo para la optimizacion de parametros continuos via simulacion basado en un algoritmo genetico hibrido*. Tesis de Magster no publicada, Escuela de Ingenieria, Pontificia Universidad Catolica de Chile, Chile.

Pannirselvam, G., Ferguson, L., Ash, R. & Siferd, S. (1999). Operations management research: an update for the 1990s. *Journal of Operations Management*(n. 18), 95–112.

Pasupathy, R., & Henderson, S. (2006). A testbed of simulation-optimization problems. *Proceedings of the 2006 Winter Simulation Conference*, 255–263.

Pichitlamken, J., Nelson, B. & Hong, L. (2006). A sequential procedure for neighborhood selection-of-the-best in optimization via simulation. *European Journal of Operational Research, 173*, 283–298.

Rechenberg, I. (1965). *Cybernetic solution path of an experimental problem*. Royal Aircraft Establishment, Farnborough p. Library Translation 1122.

Rinott, Y. (1978). On two-stage selection procedures and related probability-inequialities. *Communications in Statistics, A7*(8), 799–811.

Robbins, H., & Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, *22*(n. 3), 400–407.

Robinson, S. (1996). Analysis of sample-path optimization. *Mathematics of Operations Research, 21*, 513–528.

Rogers, P. (2002). Optimum-seeking simulation in the design an d control of manufacturing systems: Experience with OptQuest for Arena. *Proceedings of the 2002 Winter Simulation Conference*, 1142–1150.

Sabuncouglu, I., & Hatip, A. (2005). The turkish army uses simulation to model and optimize its fuel-supply system. *Interfaces*, *35*(6), 474–482.

Schaffer, J., Caruna, R., Eshelman, L. & Das, R. (1989). A study of control parameters affecting online performance of genetic algorithms for function optimization. En J. Schaffer (Ed.), *Proceedings of the 3rd international conference on genetic algorithms and their applications* (pp. 51–60). Morgan Kaufmann Publishers.

Schwefel, H. (1977). *Numerische optimierung von computer-modellen mittels der evolutionsstrategie*. Switzerland: Birkhäuser.

Schwefel, H. (1987). Collective phenomena in evolutionary systems. *Preprints of the 31st Annual Meeting of the International Society for General System Research, 2*, 1025–1033.

Shafer, S., & Smunt, T. (2004). Empirical simulation studies in operations management: Context, trends, and research opportunities. *Journal of Operations Management*(n. 22), 345–354.

Shruben, L., & Cogliano, V. (1987). An experimental procedures for simulation response surface model identification. *Communications of the Association for Computing Machinery*, *30*, 716–730.

Suri, R. (1989). Perturbation analysis: The state of the art and research issues explained via the GI/G/1 queue. *Proceedings of the IEEE*, 114–137.

Tsutsui, S., Ghosh, A., Corne, D. & Fujimoto, Y. (1997). A real coded genetic algorithm with an explorer and an exploiter populations. *Proceedings of the 7th International Conference on Genetich Algorithms*, 238–245.

Ville, M., & Kai, V. (2008). Improving maintenance decision making in the finnish air force through simulation. *Interfaces*, *38*(3), 187–201.

Whitley, D., Rana, S., Dzubera, J. & Mathias, K. (1996). Evaluating evolutionary algorithms. *Artificial Intelligence, 86*, 245–276.

Zinger, A., & Nelson, B. (1958). On the choice of the best amongst three normal populations with known variances. *Biometrika*, *45*, 436–445.

ANEXO A. PROBLEMA DE ABASTECIMIENTO DE COMBUSTIBLE

Las siguientes tablas presentan las condiciones utilizadas para el sistema 1 utilizado como problema de optimización. La tabla A.1 muestra las características de cada una de las estaciones con la capacidad de acopio de los distintos tipos de combustibles en cada una de ellas.

TABLA A.1. Capacidad Tanques (m^3)

Estación	Jet	Diesel	Gasolina
A	45.000	0	0
B	0	15.000	12.500
C	20.000	0	0
D	0	17.500	0
E	12.500	8.750	10.000
F	20.000	15.000	7.500
G	6.250	7.500	6.250
H	5.000	6.250	5.000
I	5.000	15.000	10.000
J	0	3.750	3.750
K	10.000	5.000	2.500
L	7.500	7.500	7.500
M	12.500	7.500	5.000
N	7.500	5.000	5.000

La tabla A.2 muestra las características de la demanda utilizada para resolver el problema, con la proporción de las demandas en cada estación.

TABLA A.2. Parámetros de la demanda en cada estación

Estación	Demanda	Jet	Diesel	Gasolina
F	3	0,45	0,35	0,2
G	2,5	0,45	0,35	0,2
H	2	0,45	0,35	0,2
I	1	0,45	0,35	0,2
J	2,5	0,5	0,5	0
K	2,27	0,45	0,35	0,2
L	2,86	0,45	0,35	0,2
M	2	0,45	0,35	0,2
N	2,5	0,45	0,35	0,2