Estudio de Algunos Aspectos Teóricos de los Algoritmos Genéticos

Tesis que para obtener el grado de Maestro en Inteligencia Artificial
Presenta

María Margarita Reyes Sierra
Maestría en Inteligencia Artificial
Universidad Veracruzana

Asesores:
Dr. Carlos A. Coello Coello
Dr. Onésimo Hernández Lerma
CINVESTAV-IPN

Revisores:
Dr. José Rigoberto Gabriel Argüelles
Dr. Manuel Martínez Morales
Universidad Veracruzana

Octubre, 2002

Resumen

En este trabajo se hace un estudio de los resultados teóricos que se han obtenido hasta el momento relacionados principalmente con la convergencia de los Algoritmos Evolutivos, en particular, de los Algoritmos Genéticos (AGs).

Los modelos estudiados son principalmente aquellos basados en cadenas de Markov.

Se presenta la demostración de la convergencia del Algoritmo Genético Elitista (AGE), realizada por Rudolph en 1994. En dicho trabajo, se demuestra que la matriz de transición de la cadena de Markov correspondiente al Algoritmo Genético Simple (AGS) es primitiva, condición suficiente para concluir que tal algoritmo no es capaz de converger al óptimo de la función objetivo. Sin embargo, al construir la matriz de transición correspondiente al AGE se muestra que ésta es reducible, por lo que este algoritmo sí es capaz de converger al óptimo de la función.

Posteriormente, se extiende el modelo correspondiente con el fin de estimar el tiempo esperado de convergencia de dicho algoritmo. Se detallan y ejemplifican las características generales de la matriz de transición del AGE mencionadas por Rudolph en 1994. El modelo desarrollado es usado para predecir el tiempo esperado de convergencia de un AG con parámetros mínimos. Los resultados obtenidos son comparados con los que presentan los experimentos correspondientes.

Por otra parte, se estudia la convergencia de los algoritmos evolutivos multiobjetivo. Se presentan y discuten los algoritmos de este tipo cuya convergencia ha sido demostrada. Después de presentar tales demostraciones, se discute la analogía entre tales algoritmos y los usados en la práctica, proponiendo como resultado nuevos esquemas de algoritmos que se apegan un poco más a aquellos que se usan actualmente, pero cuya convergencia no está asegurada.

Dedicatorias

A mis padres Margarita y Roberto por haberme dado la vida y la fuerza suficiente para vivir.
A ti madre por tu infinito amor, por hacer tuyos mis desvelos y por el gran entusiasmo con el que siempre me has apoyado en todas las facetas de mi vida.
A ti padre por haber sembrado en mí el instinto de lucha cotidiana e incansable por ser siempre mejor cada día y por siempre hacerme ver las dificultades de la vida como retos a mi capacidad humana e intelectual.
A ambos porque a ustedes debo todo lo que soy.

A mi esposo Efrén, porque su presencia cambió mi vida llenándola con la felicidad que brinda el verdadero amor.

Agradecimientos

A mi esposo Efrén, porque caminar por la vida es mejor y más facil desde que voy de su mano. Gracias por aceptarme tal como soy, apoyarme y darme la dicha de vernos crecer juntos cada día más.

A mis padres, por sus palabras de ánimo y apoyo. Gracias por estar siempre pendientes de mí.

A mi amigo, maestro y sobrino, Julio Aguilar, por su amistad y apoyo incondicional. Gracias por todas las horas de estudio que me dedicaste y que hicieron de mí la matemática que soy.

A mis tíos María Luisa Ortiz y Jorge Sierra, por ser como mis segundos padres y darme a las tres mejores hermanas que pude tener: Ale, Gina y Mary. Gracias por estar siempre ahí.

A mis hermanas de alma: Alejandra Zurita y Anel Klée.

Al Dr. Carlos A. Coello Coello, por darme la oportunidad de trabajar con él, por la amistad que me ha brindado y por toda la confianza que ha depositado siempre en mí.

Al Dr. Onésimo Hernández Lerma, por abrir un espacio en su apretada agenda de trabajo para escucharme y asesorarme en la realización de este trabajo.

Al Laboratorio Nacional de Informática Avanzada (LANIA), por el apoyo y todas las facilidades recibidas durante la realización del primer año de la Maestría en Inteligencia Artificial. Gracias a todos los amigos que encontré ahí y que me enseñaron tantas cosas.

Agradezco el apoyo para la realización de esta tesis de maestría proporcionado por CONACyT a través de una beca terminal proveniente del proyecto titulado "Nuevos Paradigmas en Optimización Evolutiva Multiobjetivo" (Ref. 34201-A) cuyo responsable es el Dr. Carlos A. Coello Coello.

Índice general

Resumen **3**

Introducción **9**

1. Computación Evolutiva **11**
1.1. Antecedentes Históricos 11
1.2. Algoritmos Evolutivos 14
1.3. Paradigmas 16
1.3.1. Estrategias Evolutivas 16
1.3.2. Programación Evolutiva 17
1.3.3. Algoritmos Genéticos 19
1.4. Conceptos Básicos 20
1.4.1. Fundamentos Biológicos 20
1.4.2. Conceptos de Computación Evolutiva 21

2. Elementos de Probabilidad **27**
2.1. Definiciones Básicas 27
2.1.1. Espacios de Probabilidad 27
2.1.2. Variables Aleatorias 29
2.2. Cadenas de Markov Finitas 31
2.2.1. Definiciones Básicas 31
2.2.2. Clasificación de Estados y Cadenas 35
2.2.3. Cadenas Absorbentes 36

3. Algoritmo Genético Simple **39**
3.1. Estudios de Convergencia 39
3.1.1. Algoritmo Genético sin Elitismo 39
3.1.2. Algoritmo Genético Elitista 41

3.2. Condiciones Mínimas de Convergencia 43

4. Tiempo de Convergencia **45**
4.1. Antecedentes 45
4.1.1. Modelo Basado en Convergencia de Alelos 45
4.1.2. Modelo Basado en Distancias de Hamming 48
4.2. Modelo Basado en Cadenas de Markov 51
4.2.1. Estudio de la Matriz del AGE ($\mathbf{P}^{+}$) 52
4.2.2. Tiempo Esperado de Convergencia 61
4.2.3. Experimentos 64
4.3. Conclusiones 68

5. Algoritmos Evolutivos Multi-Objetivo (AEMO) **71**
5.1. Introducción 71
5.2. El Primer AEMO 72
5.3. AEMO's de Primera Generación 73
5.4. AEMO's de Segunda Generación 77

6. Teoría de AEMO **83**
6.1. Espacios Parcialmente Ordenados 83
6.2. Primera Generación 84
6.2.1. Nichos 86
6.3. Segunda Generación 92
6.3.1. Elitismo Mediante el Uso de una Población Externa . . 92
6.3.2. Elitismo $\mu + \lambda$ 97

Conclusiones y Trabajo Futuro **103**

Introducción

Existen problemas de optimización cuyos espacios de búsqueda son tan grandes que los algoritmos clásicos más eficientes disponibles para resolverlos requieren un tiempo exponencial. Es precisamente en estos casos en los que las *heurísticas* tienen especial relevancia. La técnicas evolutivas, como *heurísticas* en sí, han demostrado su capacidad para encontrar soluciones casi óptimas en dicho tipo de problemas. Sin embargo, por su misma naturaleza, su comportamiento no es del todo conocido, es decir, no se conocen a ciencia cierta las condiciones bajo las cuales tendrán éxito o fracaso. Este es el caso, en particular, de los Algoritmos Genéticos (AG) sobre los cuales trata fundamentalmente el presente trabajo de tesis.

El AG es un técnica *heurística* basada en la evolución natural, cuya aplicación en el campo de la optimización surge de la presencia de la selección natural en dicha teoría, que impone la supervivencia del más apto. En sus inicios, el AG (llamado ahora "clásico") fue aplicado a problemas de optimización de funciones con un solo objetivo. En 1994, fue publicada la demostración de la convergencia de dicho algoritmo al óptimo global de la función en cuestión, bajo ciertas condiciones. De esta manera, a pesar de que existen múltiples variantes del AG clásico, con dicho trabajo se pudo tener una garantía del éxito de esta técnica en su versión más simple.

Sin embargo, dado que es común enfrentarnos en la vida cotidiana a problemas en los que nos interesa optimizar más de un objetivo (p. ej. repartir el mayor número de objetos con el menor costo posible), como era de esperarse, se trabajó en numerosas extensiones del AG clásico para dar lugar a lo que ahora se conoce como AG Multiobjetivo. Actualmente existen también múltiples variantes de este nuevo algoritmo, y se han hecho algunos esfuerzos por modelarlo teóricamente, aunque éstos han sido de alcance muy limitado hasta ahora.

En el presente trabajo, se lleva a cabo un estudio de algunos aspectos teóricos relacionados con la convergencia de los algoritmos evolutivos para optimización, en particular, de los AGs.

En el capítulo 1 se presenta una introducción a la Computación Evolutiva a través de sus orígenes y se proporcionan algunos conceptos básicos. Dado que los modelos teóricos que se abordan en este trabajo están principalmente basados en Cadenas de Markov, en el capítulo 2 se verán los elementos de probabilidad necesarios para introducir tal herramienta teórica.

En el capítulo 3 se estudia el AG clásico o simple; se introduce el modelo teórico correspondiente y se demuestra su convergencia. Una vez demostrada la convergencia del AG clásico, en el capítulo 4 se extiende el modelo teórico del capítulo 3 para estudiar la posibilidad de estimar el tiempo de convergencia necesario de dicho algoritmo.

En el capítulo 5 se presentan los conceptos básicos de la optimización multiobjetivo así como los principales algoritmos evolutivos multiobjetivo desarrollados hasta el momento. Finalmente, en el capítulo 6 se estudian los modelos teóricos correspondientes con los que se cuenta hasta ahora.

Capítulo 1

Computación Evolutiva

1.1. Antecedentes Históricos

La Teoría del Creacionismo dice que Dios creó el cielo y la tierra y todas las especies que en ella habitan, cada una por separado. El descontento de algunos científicos con esta teoría dio lugar al desarrollo de los principios que fungen como los orígenes de la Computación Evolutiva.

A mediados del siglo XIX el zoólogo francés Lamarck creó su propia teoría de la evolución en la que argumentaba que las características adquiridas por un organismo como resultado de un proceso de adaptación a lo largo de su vida son heredadas a las siguientes generaciones [34]. El científico alemán August Weismann demostró que la teoría de Lamarck estaba equivocada: cortó la cola de un grupo de ratas por varias generaciones y observó que la longitud de la cola de las nuevas generaciones no se veía afectada. Weismann formuló una teoría denominada del *germoplasma*, según la cual el cuerpo se divide en células "germinales" que pueden transmitir información hereditaria y en células "somáticas", que no pueden hacerlo. Para Weismann, la selección natural era lo único capaz de alterar la composición genética de un organismo [54].

En 1859, Charles Darwin publicó un libro titulado: "El origen de las especies" [11] en el que argumentó que la similitud entre padres e hijos se debe a la herencia de ciertas características y que los cambios que se observan de una generación a otra tienen como fin hacer a los nuevos organismos más aptos para sobrevivir.

En la época de Darwin, estaba de moda una teoría llamada de la recom-

Figura 1.1: August Weismann.

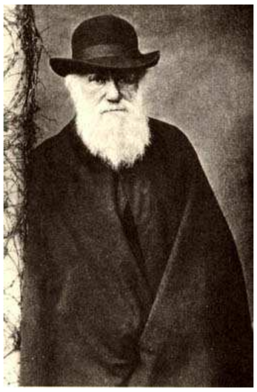

Figura 1.2: Charles Robert Darwin.

Figura 1.3: Johann Gregor Mendel.

binación según la cual los padres heredaban a los hijos ciertas características que se mezclaban o recombinaban de alguna manera en ellos, pero no explicaba la aparición en nuevos individuos de características que no habían sido observadas en sus ancestros. Esta teoría quedó desechada cuando el monje austriaco Gregor Mendel descubrió, como resultado de experimentos con chícharos, tres leyes básicas que gobiernan la herencia: la Ley de Segregación, la Ley de Independencia y la Ley de la Uniformidad. Esta última establece que las características heredadas de padres a hijos depende de si los genes de los padres son *dominantes* o *recesivos* [37].

Posterior al trabajo de Mendel, el botánico danés Hugo De Vries redescubrió sus leyes de la herencia pero además, motivado al descubrir una flor roja entre una gran cantidad de flores amarillas, desarolló la teoría de las mutaciones espontáneas que si bien no era del todo correcta ha servido para complementar la teoría evolutiva de Darwin. Según De Vries, los cambios en las especies no eran graduales sino más bien abruptos y, de hecho, aleatorios [53].

Hoy en día, se conoce como Neo-Darwinismo al paradigma resultado de unir las ideas de Darwin, Weismann y Mendel. La teoría Neo-Darwiniana establece que la vida en el planeta puede ser explicada mediante los siguientes cuatro procesos:

- Reproducción
- Mutación
- Competencia

- Selección

Es precisamente en esta teoría en la que están inspiradas las técnicas que engloba la Computación Evolutiva.

1.2. Algoritmos Evolutivos

Aunque existen diversas variantes de técnicas consideradas evolutivas, a continuación se presenta la definición formal de un Algoritmo Evolutivo [3]:

Definición 1.2.1 *Un Algoritmo Evolutivo (AE) está definido como una 8-tupla:*

$$AE = (I, \phi, \lambda, \mu, \Omega, \Psi, s, i)$$

donde:

1. *I es el espacio de individuos.*

2. *$\phi : I \to \mathbb{R}$ denota una* **función de aptitud** *que asigna valores reales a los individuos.*

3. *λ y μ son enteros positivos; $\lambda \neq \mu$ está permitido.*

4. *Ω es un conjunto de funciones aleatorias $\omega : I^{\mu} \to I^{\lambda}$, llamadas "operadores geneéticos", que obtienen λ individuos a partir de μ. Cada elemento $\omega \in \Omega$ está controlado por algún parámetro $\theta \in \mathbb{R}$.*

5. *$s : I^{\lambda} \to I^{\mu}$ denota el* **operador de selección** *que obtiene μ individuos a partir de λ.*

6. *La función de transición $\Psi : I^{\mu} \to I^{\mu}$ describe el proceso de transformación completo de una población P mediante la aplicación de los operadores genéticos y la selección, por ejemplo, para n fijo:*

$$\Psi(P) = s(\omega_1(...(\omega_n(P)))).$$

7. *$i : I^{\mu} \to \{falso, cierto\}$ es un criterio de terminación para el AE.*

Normalmente, la función de aptitud ϕ corresponde con la función objetivo del problema en cuestión, pero no necesariamente. Esto más bien depende del AE utilizado y del problema a resolverse. Por otra parte, mientras los operadores genéticos son siempre probabilísticos, la selección puede ser probabilística o completamente determinística.

El criterio de terminación puede variar desde algo arbitrariamente complejo hasta algo tan simple como completar un cierto número preestablecido de generaciones.

Definición 1.2.2 *Dado un AE con función de transición $\Psi : I^\mu \to I^\mu$ y una población inicial $P(0) \in I^\mu$, la secuencia $P(0), P(1), P(2), \ldots$ es llamada una secuencia de poblaciones o evolución de $P(0)$ y se obtiene iterativamente:*

$$\forall t \geq 0 : P(t+1) = \Psi(P(t))$$

La creación de $P(0)$ depende en particular de la instancia del AE. Usualmente el AE se inicializa de manera aleatoria, pero no necesariamente.

Con el fin de clasificar los operadores genéticos de acuerdo al número de individuos a los que son aplicados, se presenta la siguiente:

Definición 1.2.3 *Un operador genético $\omega : I^p \to I^q$ es llamado:*

- **asexual** $\Leftrightarrow \omega : I \to I$
- **sexual** $\Leftrightarrow \omega : I^2 \to I$
- **panmítico** $\Leftrightarrow \omega : I^p \to I$, *para algún* $p > 2$.

La mutación es un ejemplo de un operador asexual, mientras que la recombinación o cruza es típicamente sexual (es decir, que la recombinación se efectúa entre 2 padres), pero algunos AEs la extienden a una versión panmítica (o sea, que intervienen varios padres para generar uno o más hijos). En lo siguiente, los símbolos m y r serán usados para denotar los operadores de mutación y recombinación respectivamente.

El siguiente esquema representa un AE general:

Algoritmo Evolutivo

$t := 0$
inicializa $P(t) \in I^\mu$
evalúa $P(t)$

while $(i(P(t)) \neq true)$ **do**
recombina:$P'(t) := r(P(t))$
muta:$P''(t) := m(P'(t))$
evalúa: $P''(t) := \Psi(P''(t))$
selecciona: $P(t+1) := s(P''(t))$
$t := t+1$

Como ya se mencionó anteriormente, existen diversas variantes de las técnicas evolutivas. Sin embargo, éstas suelen clasificarse según los principales paradigmas definidos dentro de la Computación Evolutiva y que serán descritos en la siguiente sección:

- Programación Evolutiva
- Estrategias Evolutivas
- Algoritmos Genéticos

1.3. Paradigmas

1.3.1. Estrategias Evolutivas

Las Estrategias Evolutivas (EEs) son un desarrollo conjunto de Bienert, Rechenberg y Schwefel, quienes hicieron el trabajo preliminar dentro de esta área en los 1960's en la Universidad Técnica de Berlín en Alemania [3]. Las primeras aplicaciones de las EEs fueron de carácter experimental y se efectuaron en el área de hidrodinámica.

La versión original de una EE, denominada (1+1)-EE, creaba un padre y a partir de él (mediante un operador de mutación) se producía un solo hijo. De los dos se conservaba sólo al mejor y así sucesivamente.

De manera general, la población de una EE puede constar de μ individuos. En la actualidad se usa toda una variedad de mecanismos de recombinación en las EEs, entre los que se cuenta tanto con operadores **sexuales** como con operadores **panmíticos**. Así pues, de manera general:
$r : I^{\mu} \rightarrow I$.

La mutación en este caso es un operador **asexual**

$$m(\vec{x}) = \vec{x} + N(0,1)$$

Figura 1.4: Hans-Paul Schwefel.

La notación $N(0,1)$ se usa para denotar la evaluación de una variable aleatoria normalmente distribuida con media cero y desviación estándar uno.

Los operadores de selección usados son completamente **determinísticos**. Schwefel introdujo una elegante notación para estos mecanismos, caracterizando el método y el número de individuos padres e hijos, respectivamente. Además, distinguió los procesos de selección:

$$\textit{selección} - (\mu + \lambda) : I^{\mu+\lambda} \rightarrow I^{\mu}$$

Este mecanismo selecciona los μ mejores individuos a partir de la unión de los padres y los hijos para formar la siguiente generación de padres.

$$\textit{selección} - (\mu, \lambda) : I^{\lambda} \rightarrow I^{\mu}$$

Este mecanismo selecciona los μ mejores individuos a partir de los hijos únicamente para formar la siguiente generación de padres.

El esquema de una EE coincide con el esquema del AE general presentado en la sección 1.2, la direrencia esá en las características de los operadores genéticos.

Las EE han sigo aplicadas a problemas de ruteo y redes, bioquímica, óptica, diseño de ingeniería y magnetismo, por ejemplo [48].

1.3.2. Programación Evolutiva

En 1964 Lawrence J. Fogel desarrolló la Programación Evolutiva (PE). Su técnica consistía principalmente en evolucionar autómatas de estado finito con el fin de que fueran capaces de predecir los símbolos que recibían [3, 17].

Figura 1.5: Lawrence J. Fogel.

Fogel usó un modelo de mutación para ir alterando tanto los estados como las transiciones de los autómatas. Sin embargo, puesto que pretendía modelar la evolución a nivel de las especies, no usó ningún operador de recombinación, pues diferentes especies no se cruzan entre sí.

El operador de mutación usado en la programación evolutiva es **asexual** y corresponde con el operador Gaussiano. Puesto que no hay recombinación, el operador de selección se aplica sobre la unión de padres e hijos: *selección* $-(\mu+\lambda)$. La selección normalmente se lleva a cabo mediante un torneo **estocástico** para decidir qué soluciones serán retenidas.

En este caso, el esquema correspondiente a la PE se ve alterado respecto al de un AE general debido a la ausencia de la recombinación:

Programación Evolutiva

$t := 0$
inicializa $P(t) \in I^{\mu}$
evalúa $P(t)$
while $(i(P(t)) \neq true)$ **do**
  muta:$P''(t) := m(P'(t))$
  evalúa: $P''(t) := \Psi(P''(t))$
  selecciona: $P(t+1) := s(P''(t))$
  $t := t+1$

Originalmente la PE fue aplicada a problemas de predicción. Actualmente, ha sido extendida para aplicaciones en las que existen problemas de optimización continua de parámetros, planeación de rutas, reconocimiento de patrones, etc. [3, 16].

Figura 1.6: Jonh H. Holland.

1.3.3. Algoritmos Genéticos

Los Algoritmos Genéticos (AGs) son muy probablemente el tipo más conocido de AE. El más antiguo predecesor de estos algoritmos surgió con el trabajo de Fraser, un biólogo que quería simular la evolución con un especial énfasis en el estudio de la epístasis, es decir, el impacto que puede tener un cierto gene sobre otro en una posición distinta [20].

Sin embargo, el AG más usual fue desarrollado por Holland, un computólogo y psicólogo de la Universidad de Michigan. Mientras los biólogos intentaban usar los AGs para simular los sistemas biológicos, Holland notó cómo el algoritmo de búsqueda natural podía ser usado para resolver problemas que ocurrían en aplicaciones prácticas [3, 26].

La mutación en los AGs es un operador **asexual** que se aplica con probabilidad p_m y que generalmente consiste en alterar parte de la representación del individuo.

Por otra parte, la recombinación es un operador **sexual** que, con probabilidad p_c, escoge dos individuos padres y los recombina para dar lugar a dos individuos nuevos (aplicando dos veces el operador). El operador de recombinación es el operador de búsqueda más importante del AG.

Al igual que en la Programación Evolutiva el operador de selección usado en este caso es **probabilístico**.

Como en el caso de las EEs, el esquema de un AG coincide con el esquema del AE general presentado en la sección 1.2, la direrencia esá en las características de los operadores genéticos.

Los AG se han aplicado en problemas de optimización, planeación de

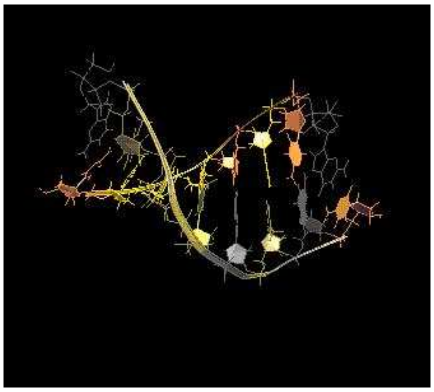

Figura 1.7: Acido Desoxirribonucleico.

movimientos, bases de datos, reconocimiento de patrones, etc. [23].

1.4. Conceptos Básicos

1.4.1. Fundamentos Biológicos

Para comenzar, recordemos que todos los seres vivos estamos compuestos del material genético conocido como **ADN** (Acido Desoxirribonucleico, figura 1.7). Dentro de las células existen cadenas de ADN que son responsables de la transmisión genética; estas cadenas reciben el nombre de **cromosomas**. Si una célula contiene un solo cromosoma o conjunto de cromosomas se le llama **haploide**, y se le denomina **diploide** en el caso de que contenga 2 copias de cada cromosoma.

Un **gene** es una sección de ADN que codifica una cierta función y sólo puede ocupar un cierto lugar dentro del cromosoma. A los diferentes valores que un gene puede tomar se les da el nombre de **alelos**.

Reciben el nombre de **gametos** aquellas células que llevan información genética de los padres con el fin de llevar a cabo la reproducción sexual. Al conjunto total de genes de un organismo se le llama **genoma**.

La reproducción sexual consiste de manera general en la recombinación o cruza de los genes de uno (asexual) o dos (sexual) padres para dar lugar al genoma de un nuevo individuo. Durante este proceso eventualmente ocurren errores de copiado conocidos como **mutaciones**.

Ahora bien, un **individuo** es el elemento básico de una población, de man-

1	0	1	1	0	0	1	0	0

Figura 1.8: Representación binaria de un cromosoma. En CE cada cromosoma corresponde con un individuo.

era que una **población** es un conjunto de individuos capaces de relacionarse e interactuar juntos. Cada individuo está determinado por su composición genética, la cual recibe el nombre de **genotipo**. Posteriormente, el genotipo da lugar a los rasgos específicos u observables del individuo, los cuales reciben el nombre de **fenotipo**.

Un individuo se desarrolla dentro de un cierto **ambiente** y este último a su vez actúa sobre el individuo alterando su capacidad de reproducirse, mejor conocida como **aptitud**. Con base en la aptitud de cada individuo, el proceso de **selección** determinará cuáles son los individuos que se reproducirán para dar lugar a nuevas generaciones.

A través de las generaciones, las frecuencias de los alelos correspondientes a los individuos de una población van cambiando como resultado del proceso de selección. Sin embargo, este cambio puede también ser un resultado de mutaciones o del azar; a este fenómeno se le llama **desvío genético**.

Un **nicho ecolólogico** se caracteriza por estar constituído por organismos de la misma especie, es decir, individuos similares que se reproducen entre sí y que además comparten la misma estrategia de supervivencia. Dos especies que ocupan nichos diferentes pueden coexistir de una manera estable. Sin embargo, dos especies que ocupan el mismo nicho ecológico compiten hasta que la más débil termina por extinguirse.

1.4.2. Conceptos de Computación Evolutiva

Dado un problema a resolver, en Computación Evolutiva (CE) un **cromosoma** se representa con una estructura de datos que codifica los parámetros de una posible solución a dicho problema.

Cada cromosoma corresponde a un **individuo** de la población. Los cromosomas usualmente son representados por cadenas binarias (figura 1.8). Dado que un **gene** es una subsección de un cromosoma, codificará el valor de un solo parámetro (figura 1.9).

Así pues, el **genotipo** corresponde con la codificación (digamos binaria) del cromosoma y el **fenotipo** con la decodificación de éste, ver figura 1.10.

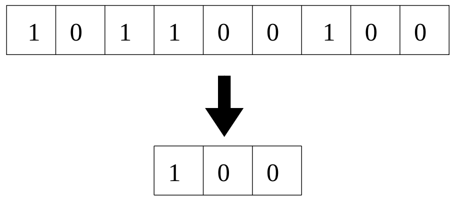

Figura 1.9: Un gene es una subcadena del cromosoma.

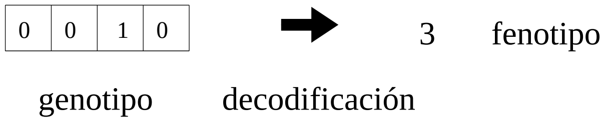

Figura 1.10: El genotipo determina directamente el fenotipo.

Los **alelos** son los valores posibles que puede tomar cada posición genética. Por ejemplo, si la codificación es binaria, un alelo puede valer 0 o 1 (figura 1.11).

La **aptitud** de un individuo es un valor que se le asigna y que denota la calidad de éste con respecto a los demás. Por ejemplo, el valor correspondiente de la función objetivo.

Se llamará **generación** a la creación de una población nueva a partir de la existente, previo cálculo de aptitudes.

En el caso en el que un individuo puede recombinarse con cualquier otro de la población, ésta recibe el nombre de **población panmítica**.

Como ya se había mencionado antes, la **epístasis** se refiere el impacto que puede tener un cierto gene sobre otro en una posición distinta. Cuando un sistema tiene poca epístasis (o ninguna) es trivial encontrar su solución. Sin embargo, en el caso contrario, el problema puede resultar bastante difícil o incluso imposible de resolver para un AE. A estos problemas se les ha dado

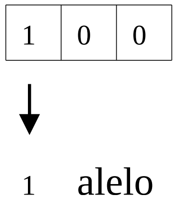

Figura 1.11: Un alelo es el valor que toma el gene.

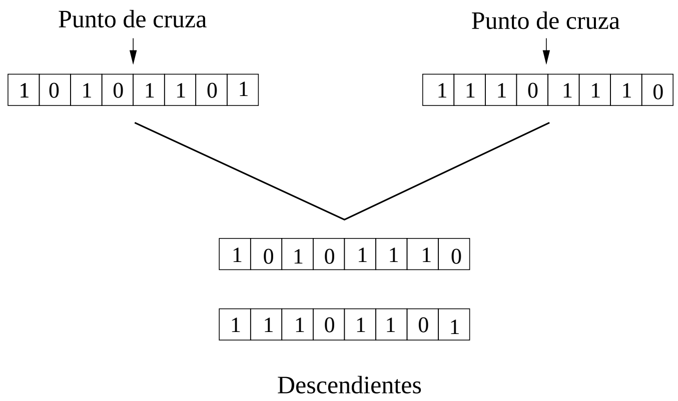

Figura 1.12: Cruza de un punto.

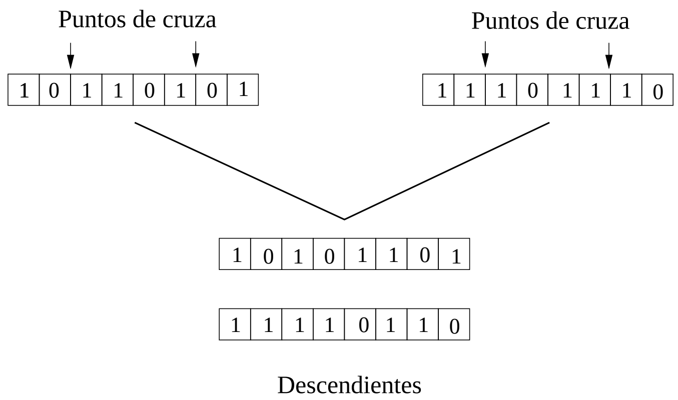

Figura 1.13: Cruza de 2 puntos.

el nombre de **deceptivos**.

En cuanto a la selección, las técnicas usadas en los Algoritmos Genéticos, por ejemplo, pueden ser:

- **Probabilísticas** (de acuerdo a la aptitud de cada individuo [26]). Ejemplos: la ruleta [31], sobrante estocástico [5, 6], universal estocástica [4], muestreo determinístico [31].

- **Mediante torneo** (comparaciones directas de los individuos). Se tienen dos versiones: determinística y probabilística [55, 6].

- **De estado uniforme**. Se usa en AGs en los que sólo unos cuantos individuos son reemplazados en cada generación [56].

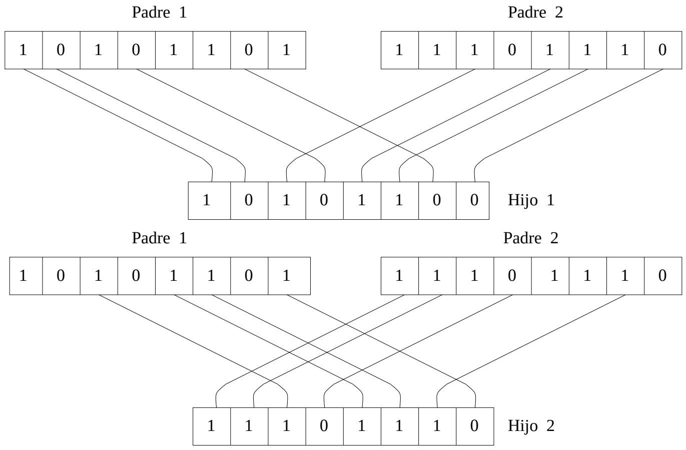

Figura 1.14: Cruza Uniforme. Cada padre aporta un gene con una probabilidad dada (de 0.5 en este caso).

Los operadores de reproducción usados en CE son clasificados en tres grupos:

- **Cruza**. Forma un nuevo individuo combinando los cromosomas de los padres. Ejemplos:
  - Cruza de n-puntos. Los padres intercambian parte de su cadena cromosómica para generar una nueva alternándose basados en los puntos de cruza (figuras 1.12, 1.13).
  - Cruza uniforme. Los padres aportan cada uno de sus alelos con una probabilidad dada para dar lugar a una nueva cadena (figura 1.14).
- **Mutación**. Obtiene un nuevo cromosoma alterando los genes del cromosoma padre. Ejemplo:
  - Mutación uniforme. Se va recorriendo la cadena y con una cierta probabilidad se altera el gene en turno (figura 1.15).
- **Reordenamiento**. Cambia el orden de los genes del cromosoma padre. Ejemplo:
  - Inversión. Los genes entre dos puntos elegidos al azar son invertidos (figura 1.16).

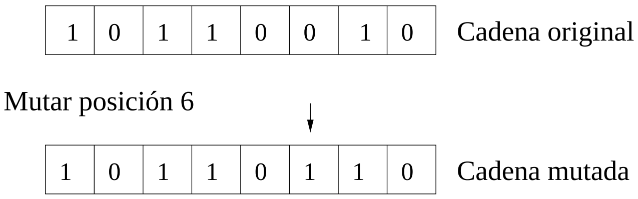

Figura 1.15: La mutación consiste en alterar el valor del gene.

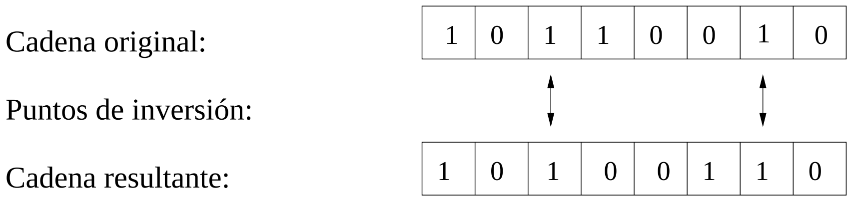

Figura 1.16: Ejemplo de inversión.

Se ha considerado que los operadores de cruza ayudan a **explotar** el espacio de búsqueda; es decir, a atravesar el espacio de búsqueda mediante movimientos pequeños en la dirección que según la información obtenida de los individuos parezca promisoria. Por otra parte, los operadores de mutación colaboran a **explorar** el espacio de búsqueda, es decir, a realizar saltos a regiones no explotadas con el fin de localizar mejores regiones y a la vez evitar quedar atrapados en óptimos locales.

En concordancia con la selección natural de los más aptos, en los AE existe un mecanismo que permite que los individuos con mejor aptitud pasen intactos durante el proceso de generación de una nueva población; este mecanismo recibe el nombre de **elitismo**. El elitismo asegura que la mejor solución encontrada hasta el momento no sea perdida a causa del proceso evolutivo. En un capítulo posterior se demostrará que el elistimo es necesario para la convergencia del AG.

Capítulo 2

Elementos de Probabilidad

El análisis que se presentará de los Algoritmos Genéticos tiene como base la teoría de la probabilidad, específicamente las cadenas de Markov finitas, por lo que en este capítulo se proporcionan los preliminares necesarios para dicho desarrollo.

2.1. Definiciones Básicas

El material que se presenta en esta sección fue tomado de las referencias [42, 7].

2.1.1. Espacios de Probabilidad

Definición 2.1.1 *Un conjunto $\mathcal{A} \neq \emptyset$, cuyos elementos son subconjuntos de un conjunto Ω, es llamado una $\sigma -$ álgebra si:*

(i) $A \in \mathcal{A} \Rightarrow A^c \in \mathcal{A}$, *donde* $A^c = \Omega - A$ *denota el complemento del conjunto* A, *y*

(ii) $\bigcup_{n=1}^{\infty} A_n \in \mathcal{A}$ *para cada sucesión* $(A_n)_{n\in\mathrm{IN}} \subset \mathcal{A}$.

Por ejemplo, sea $\Omega = \{a, b, c, d\}$ y consideremos:

$\mathcal{A}_1 = \{\emptyset, \Omega, \{a\}, \{b, c, d\}\}$
$\mathcal{A}_2 = \mathcal{A}_1 \cup \{\{b\}, \{a, b\}, \{c, d\}, \{a, c, d\}\}$ y
$\mathcal{A}_3 = \{\emptyset, \Omega, \{b, c, d\}\}$

Entonces $\mathcal{A}_1$ y $\mathcal{A}_2$ son $\sigma - \acute{a}lgebras$, mientras que $\mathcal{A}_3$ no lo es.

Definición 2.1.2 *El par $(\Omega, \mathcal{A})$, en donde $\mathcal{A}$ es una $\sigma - \acute{a}lgebra$ de subconjuntos de Ω, es llamado un* **espacio medible**. *Una función ν que asigna un número $\nu(A) \in [0, \infty]$ a cada subconjunto $A \in \mathcal{A}$, es llamada una* **medida** *sobre $(\Omega, \mathcal{A})$ si $\nu(\emptyset) = 0$ y $\nu(\cup_{n=1}^{\infty} A_n) = \sum_{n=1}^{\infty} \nu(A_n)$ para cada sucesión $(A_i)_{i \in \mathrm{IN}}$ de subconjuntos de $\mathcal{A}$ disjuntos por pares. La tercia $(\Omega, \mathcal{A}, \nu)$ es llamada un* **espacio de medida**.

Definición 2.1.3 *Sea $(\Omega, \mathcal{A})$ un espacio medible y P una medida sobre $(\Omega, \mathcal{A})$ con $\mathsf{P}(\Omega) = 1$. Entonces P es llamada una* **medida de probabilidad**, *los conjuntos $A \in \mathcal{A}$ son llamados* **eventos**, *Ω es el* **espacio muestral**, *el número $\mathsf{P}(A)$ con $A \in \mathcal{A}$ es llamado la* **probabilidad del evento A**, *y la terna $(\Omega, \mathcal{A}, \mathsf{P})$ es llamado* **espacio de probabilidad**. *Si $\mathsf{P}(A) = 1$ para algún evento A, entonces se dice que el evento ocurre* **con probabilidad 1**.

Dado un experimento en particular, el conjunto de resultados posibles es un espacio muestral. Por ejemplo, si tiramos dos dados tenemos que, importando el orden:

$$\begin{aligned}\Omega = \quad & \{\{1,1\},\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{1,6\},\\ & \{2,1\},\{2,2\},\{2,3\},\{2,4\},\{2,5\},\{2,6\},\\ & \{3,1\},\{3,2\},\{3,3\},\{3,4\},\{3,5\},\{3,6\},\\ & \{4,1\},\{4,2\},\{4,3\},\{4,4\},\{4,5\},\{4,6\},\\ & \{5,1\},\{5,2\},\{5,3\},\{5,4\},\{5,5\},\{5,6\},\\ & \{6,1\},\{6,2\},\{6,3\},\{6,4\},\{6,5\},\{6,6\}\}\end{aligned}$$

En este caso podemos definir una $\sigma - \acute{a}lgebra$ $\mathcal{A}$ de subconjuntos de Ω como $\mathcal{A} = 2^{\Omega}$, es decir, el conjunto potencia de Ω. Cada uno de los resultados tiene la misma probabilidad: $\mathsf{P}(A) = \frac{1}{36}$, para cada $A \in \Omega$. Los eventos, es decir, los elementos de $\mathcal{A}$, constan de uno o más de los posibles resultados. Por ejemplo tenemos los siguientes eventos y su probabilidad:

$$\begin{array}{ll} A_1 = \{\{1,1\},\{6,4\},\{4,5\}\} & \mathsf{P}(A_1) = \frac{1}{12} \\ A_2 = \{\{4,1\}\} & \mathsf{P}(A_2) = \frac{1}{36} \\ A_3 = \{\{2,2\},\{6,3\}\} & \mathsf{P}(A_4) = \frac{1}{18} \\ A_4 = \Omega & \mathsf{P}(A_4) = 1 \end{array}$$

El evento Ω ocurre con probabilidad 1.

2.1.2. Variables Aleatorias

Definición 2.1.4 *Sean* $(\Omega, \mathcal{A})$ *y* $(\Omega', \mathcal{A}')$ *espacios medibles. La función* $X : \Omega \to \Omega'$ *es llamada una* **función medible** *si* $X^{-1}(A') \in \mathcal{A}$ *para todo* $A' \in \mathcal{A}'$.

Definición 2.1.5 *Sea* $(\Omega, \mathcal{A}, \mathsf{P})$ *un espacio de probabilidad y* $(\Omega', \mathcal{A}')$ *un espacio medible. En este caso, una función medible* $X : \Omega \to \Omega'$ *es llamada una* **variable aleatoria**. *La función* $\mathsf{P}_X : \mathcal{A}' \to [0,1] \subset \mathbb{R}$ *con* $\mathsf{P}_X(A') := \mathsf{P}(X^{-1}(A'))$ *para todo* $A' \in \mathcal{A}'$ *es llamada la* **distribución de probabilidad** *de* X.

En otras palabras, se dice que hemos definido una variable aleatoria para un experimento aleatorio cuando hemos asociado un valor generalmente numérico a cada resultado del experimento. A continuación se presentan algunos ejemplos de variables aleatorias:

- Consideremos el experimento aleatorio que consiste en lanzar tres monedas. Supongamos que a cada elemento de su espacio muestral $E = \{ccc, ccx, cxc, xcc, cxx, xcx, xxc, xxx\}$ le asignamos un número real, que es el correspondiente al número de caras.

- En el experimento aleatorio que consiste en lanzar dos dados, asignamos a cada resultado la suma de los puntos aparecidos en cada dado.

- En el experimento que consiste en elegir al azar 500 personas y medir su estatura, la función que asocia a cada persona con su talla es una variable aleatoria.

- Consideremos el experimento que consiste en elegir al azar 100 sandías de una plantación y pesarlas. La función que asocia a cada sandía con su peso es una variable aleatoria.

Si una variable aleatoria sólo toma valores enteros, ya sea, un número finito o infinito numerable de valores diremos que es discreta (los dos primeros ejemplos). Si teóricamente, puede tomar todos los valores de un intervalo de $\mathbb{R}$, diremos que es continua (los dos últimos ejemplos). En el presente trabajo consideraremos únicamente variables aleatorias discretas.

Definición 2.1.6 *Sea $g(.)$ una función medible con valores reales y sea X un variable aleatoria discreta con valores en el conjunto $\{x_1, x_2, \cdots\}$. Si*

$$\sum_i |g(x_i)| \mathsf{P}\{X = x_i\} < \infty$$

entonces

$$\mathsf{E}[g(X)] := \sum_i g(x_i) \mathsf{P}\{X = x_i\}$$

se llama **valor esperado** *de la variable aleatoria $g(X)$. Si además, $\sum_i |g^2(x_i)| \mathsf{P}\{X = x_i\} < \infty$, entonces*

$$\mathsf{V}[g(X)] := \mathsf{E}[g(X) - \mathsf{E}[g(X)]^2]$$

$$\mathsf{D}[g(X)] := \mathsf{V}[g(X)]^{1/2}$$

son llamados la **varianza** *y la* **desviación estándar**, *respectivamente, de $g(X)$.*

Lema 2.1.1 *Sean X y Y dos variables aleatorias y a y b constantes. Entonces:*

(a) $\mathsf{E}[a] = a$

(b) $\mathsf{E}[aX + b] = a\mathsf{E}[X] + b$

(c) $\mathsf{E}[X + Y] = \mathsf{E}[X] + \mathsf{E}[Y]$.

Demostración [7].

Lema 2.1.2 *Sea X una variable aleatoria:*

$$\mathsf{V}[X] := \mathsf{E}[X^2] - \mathsf{E}[X]^2$$

Demostración [7].

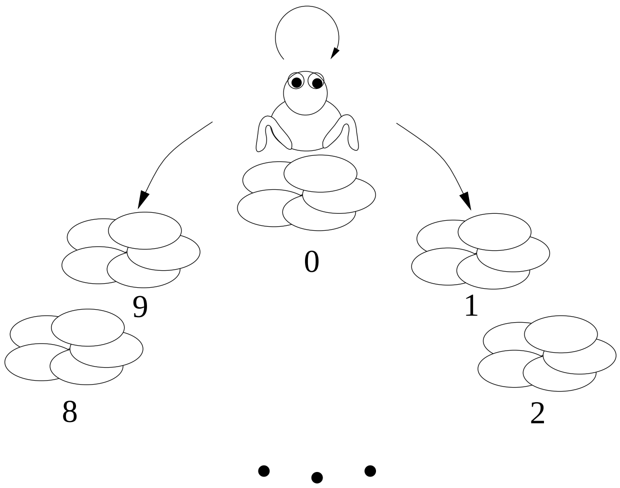

Figura 2.1: La rana puede brincar con igual probabilidad hacia las hojas adyacentes o bien brincar pero quedarse en donde está actualmente. Su estado define una cadena de Markov.

2.2. Cadenas de Markov Finitas

2.2.1. Definiciones Básicas

El material presentado en esta sección fue tomado de las referencias [45, 29].

Definición 2.2.1 *Si* $\mathcal{S} \neq \emptyset$ *es un conjunto finito y* $\{X_t : t \in \mathbb{N}\}$ *es una sucesión de variables aleatorias con valores en* $\mathcal{S}$ *con la propiedad de que:*

$$\mathsf{P}\{X_{t+1} = j | X_t = i, X_{t-1} = i_{t-1}, ..., X_0 = i_0\} = \\ \mathsf{P}\{X_{t+1} = j | X_t = i\} =: p_{ij}$$

para todo $t \geq 0$ *y para todo* $i, j \in \mathcal{S}$*, entonces la sucesión* $\{X_t : t \in \mathbb{N}\}$ *es llamada una* **cadena de Markov finita** *con espacio de estados* $\mathcal{S}$*.*

El número p_{ij} *se llama* **probabilidad de transición** *del estado* i *al estado* j *en un paso. Como estamos suponiendo que dichas probabilidades son independientes de* $t \in \mathbb{N}$*, decimos que la cadena es* **homogénea***.*

Consideremos una rana que se encuentra parada sobre una hoja en un charco. La hoja en la que está parada forma parte de un círculo de 10 hojas numeradas del 0 al 9, y la rana se encuentra actualmente en la hoja 0 (figura 2.1).

Supongamos que la rana se comporta de manera que cada minuto hace un brinco después del cual tiene tres opciones igualmente posibles: quedarse donde está o bien brincar hacia una de las dos hojas adyacentes a su hoja actual.

Sea R_t la variable aleatoria que representa la posición de nuestra rana en el tiempo t. Al inicio, nuestra rana está en la posición 0, es decir $R_0 = 0$. Posteriormente, el valor de la variable se regirá por las probabilidades de transición definidas como:

$$p_{ij} = \begin{cases} \frac{1}{3} & \text{si } i = j \text{ o si } i \text{ y } j \text{ son adyacentes,} \\ 0 & \text{de lo contrario.} \end{cases}$$

Claramente, el valor de R_t depende sólo de R_{t-1}. De esta manera, R_t determina una cadena de Markov finita homogénea.

En el caso de nuestra rana:

$$\mathcal{S} = \{0, 1, 2, 3, ..., 9\}$$

Puesto que $\mathcal{S}$ es finito, las probabilidades de transición pueden ser puestas en una **matriz de transición** $P = (p_{ij})_{i,j \in \mathcal{S}}$. Nótese que $\sum_j p_{ij} = 1$ para todo $i \in \mathcal{S}$.

Las probabilidades de transición de la variable R_t pueden ser acomodadas de la siguiente manera.

$$P = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{array} \begin{array}{c} \begin{array}{cccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{array} \\ \left(\begin{array}{cccccccccc} \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} \end{array} \right) \end{array}$$

El vector fila $\delta(t)$ con componentes $\delta_i(t) = \mathsf{P}\{X_t = i\}$ para todo $i \in \mathcal{S}$, denota la distribución de la cadena de Markov en el paso $t \geq 0$. Esta distribución puede calculare iterativamente, pues

$$\delta(t) = \delta(t-1)P = \delta(0)P^t, \text{ para todo } t \geq 0.$$

Por ejemplo, en el proceso de nuestra rana:

$$\delta(0) = \{1,0,0,0,0,0,0,0,0,0,0\}$$

$$\delta_0(0) = 1$$

De manera que:

$$\delta(1) = \delta(0)P = \{\frac{1}{3}, \frac{1}{3}, 0,0,0,0,0,0,0,0, \frac{1}{3}\}$$

$$\delta(2) = \delta(1)P = \delta(0)P^2 = \{\frac{1}{3}, \frac{2}{9}, \frac{1}{9}, 0,0,0,0,0,0, \frac{1}{9}, \frac{2}{9}\}$$

Así pues, una cadena de Markov finita homogénea está completamente determinada por su distribución inicial $\delta(0)$ y su matriz de transición P.

Definición 2.2.2

- *Una matriz $P : n \times m$ es llamada* **no negativa** *($P \geq 0$) si $p_{ij} \geq 0$ y* **positiva** *($P > 0$) si $p_{ij} > 0$ para todo $i = 1, ..., n$ y $j = 1, ..., m$.*

- *Una matriz cuadrada no negativa es llamada* **estocástica** *si la suma de cada una de sus filas es igual a 1. Así, las matrices de transición son estocásticas.*

- *Una matriz estocástica P es* **primitiva** *si*

$$\exists k \in \mathrm{IN} : P^k \text{ es positiva } (P^k > 0)$$

- *Es* **irreducible** *si*

$$\forall i,j \in \mathcal{S} : \exists k \in \mathrm{IN} : p_{ij}(k) > 0,$$

  en donde $p_{ij}(k)$ denota al elemento (i,j) de P^k. Por lo tanto, toda matriz positiva P es primitiva y toda matriz primitiva es irreducible.

- *Una matriz P es* **reducible** *(por supuesto si no es irreducible y) si puede ser acomodada en la forma:*

$$\begin{pmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{R} & \mathbf{T} \end{pmatrix}$$

  con matrices cuadradas $\mathbf{C}$ y $\mathbf{T}$.

- *Finalmente, una matriz estocástica P es* **diagonal positiva** *si cada elemento en su diagonal es positivo, es* **columna-permisible** *si al menos un elemento en cada columna es positivo y es* **estable** *si tiene filas idénticas.*

A continuación se presentan algunos resultados que serán de utilidad en capítulos posteriores.

Lema 2.2.1 *Sean P, Q y R matrices estocásticas, donde Q es positiva y R es columna-permisible. Entonces la matriz producto PQR es positiva.*

Demostración [29].

Lema 2.2.2 *Sean P, Q y R matrices estocásticas, donde Q es irreducible y P, R son diagonal positivas. Entonces el producto PQR es irreducible.*

Demostración [29].

Teorema 2.2.1 *Sea P una matriz estocástica primitiva. Entonces P^k converge cuando $k \to \infty$ a una matriz estocástica positiva estable $P^\infty = \mathbf{1}'p^\infty$, donde $\mathbf{1}'$ es un vector columna de unos y $p^\infty = p^0 \cdot \lim_{k\to\infty} P^k = p^0 P^\infty$ tiene entradas positivas y es única independientemente de la distribución inicial p^0.*

Demostración [29].

Teorema 2.2.2 *Sea $P_{n\times n}$ una matriz estocástica reducible, donde $C : m\times m$ es una matriz estocástica primitiva y R, $T \neq 0$. Entonces:*

$$P^\infty = \lim_{k\to\infty} P^k = \lim_{k\to\infty} \begin{pmatrix} C^k & 0 \\ \sum_{i=0}^{k-1} T^i R C^{k-i} & T^k \end{pmatrix} = \begin{pmatrix} C^\infty & 0 \\ R_\infty & 0 \end{pmatrix}$$

es una matriz estocástica estable con $P^\infty = \mathbf{1}'p^\infty$, donde $\mathbf{1}'$ es un vector columna de unos y $p^\infty = p^0 P^\infty$ es única independientemente de la distribución inicial y, además, p^∞ satisface: $p_i^\infty > 0$ para $1 \leq i \leq m$ y $p_i^\infty = 0$ para $m < i \leq n$.

Demostración [29].

2.2.2. Clasificación de Estados y Cadenas

Considérese una cadena de Markov con espacio de estados finito $\mathcal{S} \neq \emptyset$.

En esta sección se clasificarán los estados de una cadena de Markov de acuerdo a si es posible ir de un determinado estado a otro estado determinado [32].

Definición 2.2.3 *Decimos que el estado i* **induce** *al estado j y lo denotamos por $i \to j$ si y sólo si $p_{ij}^k > 0$ para algún $k \geq 1$. Si tenemos que $i \to j$ y $j \to i$ decimos que el estado i está* **comunicado** *con el estado j (o que los estados i, j son* **comunicantes***) y lo denotamos por $i \leftrightarrow j$.*

A partir de la definición anterior, los estados son divididos en "clases de equivalencia". Dos estados están en la misma clase de equivalencia si están "comunicados", es decir, si el proceso puede ir de un estado a otro y viceversa.

Las clases de equivalencia son clasificadas como conjuntos **ergódicos** (también llamados **recurrentes**) o conjuntos **transitorios**. De esta manera los estados pertenecientes a ellas serán llamados estados ergódicos y transitorios, respectivamente.

Para cada cadena de Markov finita debe existir siempre al menos un conjunto ergódico; sin embargo, los conjuntos transitorios no necesariamente existen.

Una vez que una cadena abandona un conjunto transitorio nunca regresará a él, mientras que una vez que entra en un conjunto ergódico nunca podrá abandonarlo.

En particular, si un conjunto ergódico contiene sólo un estado, éste recibe el nombre de estado **absorbente** pues una vez estando en él, la cadena de Markov se quedará allí por siempre. De lo anterior tenemos el siguiente:

Teorema 2.2.3 *Un estado i es absorbente si y sólo si $p_{ii} = 1$.*

Demostración [32].

Los conjuntos ergódicos pueden ser clasificados de dos maneras:

1. **Regular**: En este caso, el conjunto consta de un ciclo único de estados, por lo que no importa cuál sea el estado en el que la cadena inicie, después de un tiempo suficientemente grande la cadena podría estar en cualquier estado del conjunto (matriz de transición primitiva).

2. **Cíclico** (o **periódico**): En este caso el conjunto está dividido en d ciclos distintos de manera que dado un estado de inicio determinado, la cadena irá recorriendo los diferentes ciclos hasta regresar al ciclo en el que inició después de exactamente d pasos.

Las cadenas de Markov pueden clasificarse según si éstas contienen o no conjuntos transitorios:

I **Cadenas sin Conjuntos Transitorios**
Sin pérdida de generalidad supondremos que en este caso existe sólo un conjunto ergódico, es decir, todo el conjunto de estados de la cadena es un conjunto ergódico. Una cadena que consiste de un único conjunto ergódico es llamada una **cadena ergódica**, y puede coincidir con algunos de los siguientes dos casos:

I-A El conjunto ergódico es regular.

I-B El conjunto ergódico es cíclico.

II **Cadenas con Conjuntos Transitorios**
En este caso, la cadena se mueve hacia los conjuntos ergódicos. La probabilidad de que el proceso se encuentre dentro de un conjunto ergódico tiende a 1. En este caso nuevamente podemos clasificar este tipo de cadenas con base en las características de sus conjuntos ergódicos.

II-A Todos los conjuntos ergódicos son conjuntos unitarios. Este tipo de cadenas son llamadas **cadenas absorbentes**, pues eventualmente quedarán atrapadas en un estado absorbente.

II-B Todos los conjuntos ergódicos son regulares, pero no unitarios.

II-C Todos los conjuntos ergódicos son cíclicos.

II-D Existen tanto conjuntos ergódicos regulares como cíclicos.

2.2.3. Cadenas Absorbentes

Son de nuestro especial interés en este trabajo las cadenas absorbentes, por lo que en esta sección se presentarán algunas propiedades importantes de ellas [32].

Teorema 2.2.4 *Para cualquier cadena de Markov finita absorbente, no importa el estado en el que el proceso inicie, la probabilidad de que el proceso se encuentre en un estado absorbente después de n pasos tiende a 1 conforme n tiende a infinito.*

Demostración [32].

Es importante considerar la forma canónica de la matriz de transición de una cadena de Markov. Supongamos que tenemos s estados transitorios y $r-s$ estados ergódicos y que reunimos a todos los conjuntos transitorios y a todos los conjuntos ergódicos, la forma es la siguiente:

$$P = \begin{array}{c c} \begin{array}{cc} r-s & s \end{array} & \\ \begin{pmatrix} S & O \\ R & Q \end{pmatrix} & \begin{array}{l} \}r-s \\ \}s \end{array} \end{array}$$

La región O consiste completamente de ceros. La matriz $Q_{s\times s}$ representa a la cadena mientras ésta se encuentre en estados transitorios, la matriz $R_{s\times(r-s)}$ representa la transición de estados transitorios a estados ergódicos y la matriz $S_{(r-s)\times(r-s)}$ representa la cadena una vez que ha alcanzado un estado ergódico.

Si consideramos una cadena absorbente, tenemos que por definición $S = I_{(r-s)\times(r-s)}$, así su forma canónica es:

$$P = \begin{array}{c c} \begin{array}{cc} r-s & s \end{array} & \\ \begin{pmatrix} I & O \\ R & Q \end{pmatrix} & \begin{array}{l} \}r-s \\ \}s \end{array} \end{array}$$

Por el Teorema 2.2.2 podemos ver que las potencias de Q tienden a O.

Teorema 2.2.5 *Para cualquier cadena de Markov absorbente, $I-Q$ tiene una inversa y,*

$$(I-Q)^{-1} = I + Q + Q^2 + \cdots = \sum_{k=0}^{\infty} Q^k$$

Demostración [32].

Definición 2.2.4 *Para una cadena de Markov absorbente definimos la* **matriz fundamental** *como $N = (I-Q)^{-1}$.*

Definición 2.2.5 *Definimos* $\mathbf{n}_j$ *como una función cuyo valor es el número total de veces que el proceso está en el estado* s_j*. (Esta definición sólo es válida para estados transitorios* s_j*). Definimos a* $\mathbf{u}_j^k$ *como una función que toma el valor de 1 si el proceso está en el estado* s_j *en el paso* k*, y 0 de otra manera.*

Es fácil ver que:

$$\mathbf{n}_j = \sum_{k=0}^{\infty} \mathbf{u}_j^k$$

Sea $\mathbf{T}$ el conjunto de estados transitorios de la cadena de Markov, si denotamos con $\mathsf{E}_i[\mathbf{n}_j]$ al valor esperado de $\mathbf{n}_j$ suponiendo que el proceso inicia en el estado s_i, tenemos el siguiente:

Teorema 2.2.6 $\{\mathsf{E}_i[\mathbf{n}_j]\} = N$

Demostración [32].

Definición 2.2.6 *Sea* $\mathbf{t}$ *una función cuyo valor está dado por el número de pasos (incluyendo la posición original) en los que el proceso se encuentra en un estado transitorio.*

Si el proceso inicia en un estado ergódico entonces $\mathbf{t} = 0$. Si el proceso inicia en un estado transitorio, entonces $\mathbf{t}$ nos da el número total de pasos necesarios para alcanzar un estado ergódico. En una cadena absorbente este es el *tiempo de absorción.*

Sea ξ un vector columna cuyas entradas son todas 1.

Teorema 2.2.7 $\{\mathsf{E}_i[\mathbf{t}]\} = N\xi$

Demostración Es fácil ver que

$$\mathbf{t} = \sum_{s_j \in \mathbf{T}} \mathbf{n}_j$$

Así,

$$\{\mathsf{E}_i[\mathbf{t}]\} = \{\mathsf{E}_i[\sum_{s_j \in \mathbf{T}} \mathbf{n}_j]\} = \{\sum_{s_j \in \mathbf{T}} \mathsf{E}_i[\mathbf{n}_j]\} = N\xi.$$

■

Teorema 2.2.8 $\{\mathsf{V}_i[\mathbf{t}]\} = (2N - I)N\xi - (N\xi)^2$

Demostración [32].

Capítulo 3

Algoritmo Genético Simple

Los Algoritmos Genéticos se usan a menudo para resolver problemas de optimización del tipo: $max\{f(b)|b \in \mathbb{B}^l = \{0,1\}^l\}$ (el conjunto $\mathbb{B}^l$ contiene a todas las cadenas de ceros y unos de longitud l) suponiendo que $0 < f(b) < \infty$ para todo $b \in \mathbb{B}^l$ y $f(b) \neq const.$

En este capítulo iniciamos nuestro estudio de la convergencia de los AGs. Consideraremos principalmente dos casos, el AG **sin** elitismo, también conocido como AG Simple (AGS) y el AG **con** elitismo (AGE).

3.1. Estudios de Convergencia

3.1.1. Algoritmo Genético sin Elitismo

En [41] Rudolph modela el Algoritmo Genético Simple (AGS) mediante una cadena de Markov finita homogénea.

Cada estado i de la cadena de Markov corresponde a una posible población del AGS de tal manera que el espacio de estados es $\mathcal{S} = \mathbb{B}^{nl}$ donde n es el número de individuos de la población y l es la longitud de cada individuo. Definimos a $\pi_k^t(i)$ como el individuo k de la población i en el paso t.

Dada la naturaleza del AGS, la matriz de transición $\mathbf{P}$ que lo representa queda definida como

$$\mathbf{P} = \mathbf{CMS},$$

donde $\mathbf{C}$, $\mathbf{M}$ y $\mathbf{S}$ son las matrices de transición de los operadores de cruza, mutación y selección respectivamente.

Cuando se usa *mutación uniforme* los elementos de $\mathbf{M}$ son

$$m_{ij} = p_m^{H_{ij}}(1-p_m)^{N-H_{ij}} > 0,$$

en donde p_m es la probabilidad de mutación del AGS, H_{ij} es la distancia de Hamming entre los estados i y j, y $N = nl$ (la distancia de Hamming entre dos cadenas binarias se define como el número de posiciones en las que tales cadenas son distintas). De lo anterior concluímos que $\mathbf{M}$ *es positiva.*

Por otra parte, dado que el operador de selección lo que hace es proporcionarnos pares de individuos con fines de que ya sea que pasen intactos a la población siguiente o que con una cierta probabilidad mediante el operador de cruza puedan dar lugar a otros dos individuos nuevos, la matriz de transición de este operador lo que hace es dejar "ordenados" a los individuos tal y como se irán tomando para dar lugar a la población siguiente.

El uso de un operador de selección proporcional o de torneo [45] determina la existencia de una probabilidad estrictamente positiva de que la población quede intacta, lo cual asegura que los elementos de la diagonal s_{ii} de la matriz de transición del operador son positivos, por lo que se concluye que la matriz $\mathbf{S}$ *es columna-permisible.*

En resumen tenemos que la matriz $\mathbf{M}$ es positiva y $\mathbf{S}$ es columna-permisible. Luego, por el Lema 2.2.1, la matriz $\mathbf{P} = \mathbf{CMS}$ *es positiva y por lo tanto primitiva.*

Para poder hablar de convergencia a continuación se presenta la definición correspondiente para el AGS [41]:

Definición 3.1.1 *Sea $Z_t = max\{f(\pi_k^t(i))|k = 1, ..., n\}$ una sucesión de variables aleatorias que denotan la mejor aptitud dentro de la población representada por el estado i en el paso t. Un algoritmo genético converge al óptimo global si y sólo si:*

$$lim_{t\to\infty} P\{Z_t = f^*\} = 1 \tag{3.1.1}$$

donde $f^* = max\{f(b)|b \in \mathbb{B}^l\}$.

De esta manera entenderemos que el AGS converge al óptimo global de la función objetivo si la probabilidad de que éste se encuentre en la población tiende a 1 cuando el número de iteraciones tiende a infinito.

Así pues, dada la definición anterior y usando el Teorema 2.2.1 Rudolph demuestra que el AGS **no** converge:

Teorema 3.1.1 *El AGS con matriz de transición primitiva no converge al óptimo global.*

Demostración Sea $i \in \mathcal{S}$ cualquier estado en el que $max\{f(\pi_k^t(i))|k = 1, ..., n\} < f^*$ y $p_i(t)$ la probabilidad de que el AGS esté en tal estado i en el paso t. Claramente, $P\{Z_t \neq f^*\} \geq p_i(t) \Leftrightarrow P\{Z_t = f^*\} \leq 1 - p_i(t)$. Por el Teorema 2.2.1 la probabilidad de que el AGS esté en el estado i converge a $p_i(\infty) > 0$. Por lo tanto:

$$lim_{t\to\infty}P\{Z_t = f^*\} \leq 1 - p_i(\infty) < 1,$$

es decir, la condición (3.1.2) no se satisface. ■

El Teorema 3.1.1 muestra que dado que según el Teorema 2.2.1 la matriz de transición **P** del AGS converge a una matriz positiva, la probabilidad de estar en un estado no-óptimo es estrictamente positiva conforme el número de iteraciones se incrementa por lo que la probabilidad de permanecer en un estado óptimo no es 1 en el límite.

3.1.2. Algoritmo Genético Elitista

En [41] Rudolph argumenta que en las aplicaciones del mundo real el AGS comúnmente mantiene a través del proceso evolutivo la mejor solución encontrada hasta el momento por lo que lo correcto es modelar el AGS de tal manera.

Así pues, consideraremos ahora agregar a la población del AGS un *súper individuo* que no tomará parte en el proceso evolutivo y que por facilidad en la notación será colocado en la primera posición a la izquierda, es decir, se podrá accesar a él mediante $\pi_0(i)$. Llamaremos a esta nueva versión Algoritmo Genético Elitista (AGE).

La cardinalidad del espacio de estados de la cadena de Markov correspondiente crece ahora de 2^{nl} a $2^{(n+1)l}$ debido a que tenemos 2^l posibles *súper individuos* y por cada uno de ellos tenemos 2^{nl} poblaciones posibles.

El operador de elitismo estará representado por la matriz **E** que lo que hará será actualizar un estado de tal manera que si éste contiene un individuo mejor que su actual *súper individuo* éste será reemplazado por aquél.

En particular, sea:

$$i = (\pi_0(i), \pi_1(i), \pi_2(i), ..., \pi_n(i)) \in \mathcal{S}$$

$\pi_0(i)$ es el súper individuo de la población (estado) i. Ahora bien, sea $b = \text{argmax}\{f(\pi_k(i))|k = 1, ..., n\} \in \mathbb{B}^l$ el mejor individuo de la población i excluyendo el *súper individuo* y:

$$j \stackrel{\text{def}}{=} (b, \pi_1(i), \pi_2(i), ..., \pi_n(i)) \in \mathcal{S}$$

entonces:

$$e_{ij} = \begin{cases} 1 & \text{si } f(\pi_0(i)) < f(b) \\ 0 & \text{de otra manera.} \end{cases}$$

La nueva matriz de transición para el AGE resulta del producto de una matriz que está compuesta por 2^l matrices $\mathbf{P}$, una por cada posible *súper individuo* acomodadas de manera que entre mejor sea su súper individuo más alta será su posición, y la matriz $\mathbf{E}$ del operador de elitismo:

$$\mathbf{P}^+ = \begin{pmatrix} \mathbf{P} & & & \\ & \mathbf{P} & & \\ & & \ddots & \\ & & & \mathbf{P} \end{pmatrix} \begin{pmatrix} \mathbf{E_{11}} & & & \\ \mathbf{E_{21}} & \mathbf{E_{22}} & & \\ \vdots & \vdots & \ddots & \\ \mathbf{E_{2^l,1}} & \mathbf{E_{2^l,2}} & \cdots & \mathbf{E_{2^l,2^l}} \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{PE_{11}} & & & \\ \mathbf{PE_{21}} & \mathbf{PE_{22}} & & \\ \vdots & \vdots & \ddots & \\ \mathbf{PE_{2^l,1}} & \mathbf{PE_{2^l,2}} & \cdots & \mathbf{PE_{2^l,2^l}} \end{pmatrix}$$

La estructura mostrada de la matriz $\mathbf{P}^+$ se debe a que, como ya se mencionó, las poblaciones están ordenadas de manera descendente de acuerdo a la calidad de su súper individuo. De tal manera los espacios en blanco representan ceros puesto que no es posible pasar de un estado a otro con un súper individuo de menor calidad.

De lo anterior se concluye que $\mathbf{PE_{11}} = \mathbf{P}$ puesto que tales matrices corresponden con las poblaciones que tienen como súper individuo al óptimo f^*.

Por otra parte, haciendo las siguientes definiciones:

$$\mathbf{R} = \begin{pmatrix} \mathbf{PE_{21}} \\ \vdots \\ \mathbf{PE_{2^l,1}} \end{pmatrix} \mathbf{T} = \begin{pmatrix} \mathbf{PE_{22}} & & \\ \vdots & \ddots & \\ \mathbf{PE_{2^l,2}} & \cdots & \mathbf{PE_{2^l,2^l}} \end{pmatrix}$$

concluimos que la matriz $\mathbf{P}^+$ es reducible a la forma:

$$\mathbf{P}^+ = \begin{pmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{R} & \mathbf{T} \end{pmatrix}.$$

Como conclusión tenemos el siguiente teorema:

Teorema 3.1.2 *El AGE converge al óptimo global.*

Demostración La submatriz $\mathbf{P}$ contiene las probabilidades de transición de estados óptimos globales. Puesto que $\mathbf{P}$ es una matriz estocástica primitiva y $\mathbf{R}$, $\mathbf{T} \neq 0$, el Teorema 2.2.2 garantiza que la probabilidad de permanecer en un estado no-óptimo converge a cero. Por lo tanto la probabilidad de permanecer en un estado óptimo global converge a 1. ∎

Así pues, se ha demostrado la convergencia del AGE, es decir, de un Algoritmo Genético que usa elitismo.

3.2. Condiciones Mínimas de Convergencia

Posterior al trabajo de Rudolph, Alexandru Agapie [1] realizó un par de modificaciones a las condiciones impuestas en el modelo del AGS desarrollado en la sección anterior.

Alexandru hace notar que la condición de que la matriz del operador de mutación ($\mathbf{M}$) sea positiva es demasiado fuerte y que, de hecho, es un tanto inadecuada para los AGs comunes. Por ejemplo, supongamos que se tiene una población de tamaño 20 con individuos de longitud 20 y p_m= 0.1. Entonces la probabilidad de pasar del estado $i = (111\ldots1)$ al estado $j = (000\ldots0)$ es aproximadamente $m_{ij} \sim 10^{-400}$. Como podemos ver, tal probabilidad de transición es tan pequeña que podría ser considerada como cero.

Por tal razón, en este trabajo se demuestra que aunque la matriz de transición $\mathbf{M}$ no sea positiva la convergencia se conserva, aunque cabe mencionar que se introduce una condición a la matriz correspondiente a la cruza que, si recordamos, era arbitraria.

La principal idea del trabajo de Agapie es la siguiente: el permitir mutaciones de un solo bit es suficiente para hacer que el AGE converja puesto que mutaciones de varios bits pueden ser llevadas a cabo "por partes" mediante mutaciones de un solo bit a la vez. De esta manera, la condición de que la matriz $\mathbf{M}$ sea positiva puede ser relajada a que la matriz $\mathbf{M}$ sea irreducible.

De manera general, supongamos que un AGE tiene una matriz de transición del operador de mutación tal que sólo permite mutar como máximo un número fijo de bits, digamos T, menor que la longitud del cromosoma.

Denotemos con H_{ij} la distancia de Hamming entre las poblaciones i y j. Lo que queremos demostrar ahora es que la matriz de mutación descrita es irreducible.

Lema 3.2.1 *Sean T, $1 \leq T \leq 2^{nl}$, un entero y $\mathbf{M}$ la matriz estocástica correspondiente al operador de mutación de un AG que satisface: $m_{ij} = 0$ si $H_{ij} > T$ y $m_{ij} > 0$ de otra manera. Entonces $\mathbf{M}$ es irreducible.*

Demostración $m_{ij} > 0$ para cada par (i, j) tal que $H_{ij} \leq 1$. Ahora, sean i y j dos poblaciones con $H_{ij} = k$, $1 \leq k \leq 2^{nl}$. Es claro que puede encontrarse una cadena $(i, i_1, i_2, ..., i_{k-1}, j)$ tal que $H_{i,i_1} = H_{i_1,i_2} =, ..., = H_{i_{k-1},j} = 1$ y $m_{i,i_1} m_{i_1,i_2} ... m_{i_{k-1},j} = m_{ij} > 0$. Así, $\mathbf{M}$ es irreducible. ∎

Si recordamos, en la sección anterior se mencionó que la matriz de transición de un operador de selección proporcional es columna-permisible, pero de hecho se concluyó que es diagonal positiva. Por otra parte, el uso de una probabilidad de cruza $p_c < 1$ da a la matriz de transición la propiedad de ser también diagonal positiva [1].

En resumen, tenemos en este caso que la matriz $\mathbf{M}$ es irreducible y las matrices $\mathbf{C}$ y $\mathbf{S}$ son columna-permisibles; por tanto por el Lema 2.2.3 la matriz $\mathbf{P} = \mathbf{CMS}$ es nuevamente irreducible. Así pues, todos los resultados de la sección anterior siguen siendo válidos (recordemos que en la sección anterior la matriz $\mathbf{P}$ es primitiva y, en consecuencia, irreducible).

Capítulo 4

Tiempo de Convergencia

El problema de caracterizar el comportamiento de los AG en varios dominios es complejo puesto que varía con la aplicación así como con los parámetros de implementación.

En este capítulo se mostrará el trabajo basado en cadenas de Markov que se llevó a cabo para estimar el tiempo de convergencia de un AG.

4.1. Antecedentes

Como antecedentes a este trabajo, a continuación se presentan los modelos desarrollados por Carol A. Ankenbrandt [2] y Sushil J. Louis & Gregory J. E. Rawlins [35].

4.1.1. Modelo Basado en Convergencia de Alelos

En [2], Carol A. Ankenbrandt obtiene una cota para el tiempo de ejecución necesario para la convergencia del AG mediante una sencilla prueba de inducción matemática.

Aunque en el trabajo de Ankenbrandt se obtienen resultados para algoritmos binarios y no binarios, nos limitaremos a presentar los resultados correspondientes a los binarios. Sin embargo, cabe mencionar que los resultados para los algoritmos no binarios son análogos y representan el mismo grado de dificultad.

En este trabajo se analiza la convergencia en función de los valores que toman cada uno de los alelos por separado, es decir, se considerará que el AG ha convergido cuando cada alelo ha convergido. De esta manera, se analiza sólo el tiempo que le toma a un alelo converger.

Sea t el tiempo y P_i la proporción de alelos que han tomado el valor 1 en el tiempo $t = i$ en una posición particular j. Sea f_1 la aptitud de todos los individuos que han tomado el valor 1 en la posición j y sea f_0 la aptitud de todos los organismos que han tomado el valor 0 en esa misma posición. Sea $r = f_1/f_0$ la razón de aptitud y asumamos que se mantiene constante a través del tiempo.

De lo anterior, se tiene la siguiente relación de recurrencia:

$$P_{t+1} = \frac{f_1 P_t}{f_1 P_t + (1 - P_t) f_0} = \frac{r P_t}{1 + (r - 1) P_t}$$

A partir de la fórmula anterior se pueden obtener los términos P_1, P_2 y P_3 en términos de P_0 (omitiendo las operaciones intermedias):

$$P_1 = \frac{r P_0}{1 + (r - 1) P_0}$$

$$P_2 = \frac{r^2 P_0}{1 - P_0 + r^2 P_0}$$

$$P_3 = \frac{r^3 P_0}{1 - P_0 + r^3 P_0}$$

De lo anterior se intuye la fórmula general:

$$P_t = \frac{r^t P_0}{1 - P_0 + r^t P_0}$$

la cual será demostrada por inducción:

1. El caso base es claramente válido (P_1).

2. El siguiente paso es demostrar:

$$P_t = \frac{r^t P_0}{1 - P_0 + r^t P_0} \Rightarrow P_{t+1} = \frac{r^{t+1} P_0}{1 - P_0 + r^{t+1} P_0}$$

3. Demostración:

   Relación de recurrencia:

$$P_{t+1} = \frac{r P_t}{1 + (r - 1) P_t}$$

Sustituyendo P_t:

$$P_{t+1} = \frac{r\left[\frac{r^t P_0}{1-P_0+r^t P_0}\right]}{1+(r-1)\left[\frac{r^t P_0}{1-P_0+r^t P_0}\right]}$$

$$P_{t+1} = \frac{r^{(t+1)} P_0}{(1-P_0)+r^t P_0+(r-1)(r^t P_0)}$$

$$P_{t+1} = \frac{r^{(t+1)} P_0}{(1-P_0)+(1+r-1)r^t P_0}$$

$$P_{t+1} = \frac{r^{(t+1)} P_0}{(1-P_0)+r^{t+1} P_0}$$

De esta manera ha sido demostrado que la proporción de alelos que han tomado el valor de 1 en el tiempo t en una posición particular j puede ser calculada con la fórmula:

$$P_t = \frac{r^t P_0}{1-P_0+r^t P_0}$$

Sea t_c el tiempo necesario para que el sistema converja y P_f el valor de P_t en tal instante. Tenemos entonces:

$$P_f = \frac{r^{t_c} P_0}{1-P_0+r^{t_c} P_0}$$

Despejando t_c (se omiten las operaciones):

$$t_c = \frac{\ln\left[\frac{P_f(1-P_0)}{P_0(1-P_f)}\right]}{\ln r}$$

Así pues, asumiendo que el tamaño de la población es m procederemos a obtener el valor de t_c en el peor caso y en el caso promedio. En ambos casos se asume una convergencia con tolerancia γ de tal manera que $P_f = 1-\gamma$.

1. En el peor caso en la población inicial sólo un alelo ha tomado el valor correcto por lo que en este caso: $P_0 = 1/m$. Por lo que se obtiene:

$$t_c = \frac{\ln((m-1)^2)}{\ln r}$$

2. En el caso promedio se tiene que P_0 =0.5. Por lo tanto:

$$t_c = \frac{\ln(m-1)}{\ln r}$$

En este trabajo finalmente se aclara que para obtener una cota más real en una cierta aplicación es necesario multiplicar el orden de los valores obtenidos anteriormente por el orden correspondiente del proceso de evaluación según sea el caso.

4.1.2. Modelo Basado en Distancias de Hamming

Dado que la selección natural usa la diversidad en una población para dar lugar a la adaptación, en [35] Louis & Rawlins argumentan que ignorando los efectos de la mutación, si no existe diversidad no hay nada en lo que la selección natural pueda trabajar, por lo que usan una medida de diversidad para estimar el tiempo de convergencia necesario.

Así pues, según [35], un AG converge cuando toda la población es idéntica o cuando la diversidad es mínima. Usando un promedio de las distancias de Hamming entre todos los miembros de una población como una medida de diversidad, los autores derivan una cota superior para el tiempo necesario para llegar a un estado de diversidad mínima en el cual el AG posiblemente no pueda ya progresar de manera significativa. Sin embargo, el análisis que se va a presentar no predice de ninguna manera la calidad de la solución a la que se ha convergido.

El modelo asume un AG con selección proporcional, cruza de n-puntos y sin mutación. Sea l la longitud de los individuos de la población. El promedio de distancias de Hamming en la población inicial se puede aproximar bastante bien con un distribución normal con media h_0:

$$h_0 = l/2$$

y desviación estándar s_0:

$$s_0 = \sqrt{l}/2$$

Dado que se están ignorando los efectos de la mutación, el promedio de distancias de Hamming de una población que ya convergió es cero. Por lo tanto se procederá a analizar los efectos de la selección y la cruza.

Respecto a los efectos causados por la cruza, se tiene lo siguiente:

Lema 4.1.1 *Los operadores de cruza tradicionales, como la cruza de n-puntos, no cambian el promedio de las distancias de Hamming de una población dada.*

Demostración Se probará que el promedio de las distancias de Hamming en la generación $t+1$ es el mismo que en la generación t bajo el efecto de la cruza. Asumiendo un alfabeto binario, podemos expresar el promedio de distancias de Hamming de la población en la generación t como la suma de los promedios para cada uno de los l genes. Sea $h_{i,t}$ el promedio de Hamming del gene i:

$$h_t = \sum_{i=1}^{l} h_{i,t}$$

Entonces el promedio de Hamming en la siguiente generación es:

$$h_{t+1} = \sum_{i=1}^{l} h_{i,t+1}$$

En la ausencia de selección y mutación, la cruza sólo cambia el orden en el cual se suman las contribuciones de cada gene. Esto es:

$$h_{i,t} = h_{i,t+1}$$

Por lo tanto:

$$h_t = h_{t+1}$$

■

Habiendo visto que la cruza no afecta el promedio de Hamming se procederá a analizar el efecto de la selección. La selección es la parte de un AG que depende del dominio en cuestión. Obtener una cota superior para el tiempo de convergencia es asumir el peor caso y estimarlo, por lo que se usará la función:

$$f(x) = constante$$

esta función no contiene ningún tipo de información útil para un algoritmo de búsqueda, por lo que ningún algoritmo podrá hacerlo mejor que la búsqueda aleatoria. Sin embargo, el AG perderá diversidad y eventualmente convergerá gracias al desvío genético [35].

Así pues, una expresión para el tiempo de convergencia en este caso nos da una cota superior para el AG en cualquier función estática.

Si una población de tamaño N contiene una proporción p_i del alelo binario i, entonces la probabilidad de que se produzcan k copias del alelo i en la siguiente generación es:

$$\binom{N}{k} p_i^k (1-p_i)^{N-k}$$

Usando esta distribución se tiene que calcular la probabilidad de que ocurra una cierta frecuencia del alelo i en las generaciones siguientes. Sea $f(p,t)$ la probabilidad de que el alelo tenga una frecuencia p en la generación t, donde $0 < p < 1$, entonces[12]:

$$f(p,t) = \frac{6p_0(1-p_0)}{N}\left(1-\frac{2}{N}\right)^t$$

donde p_0 es la frecuencia del alelo i en $t=0$. La función anterior especifica la probabilidad de que un alelo no haya convergido, por lo que la probabilidad de que el valor del alelo esté fijo, es decir, que haya convergido en la generación t es:

$$\mathcal{P}(t) = 1 - f(p,t)$$

Aplicando esto a un AG y asumiendo que los alelos convergen de manera independiente, la probabilidad de que todos los alelos hayan convergido en la generación t es:

$$\mathcal{P}(t) = \left[1 - \frac{6p_0(1-p_0)}{N}\left(1-\frac{2}{N}\right)^t\right]^l$$

Así pues, la ecuación anterior nos da el tiempo de convergencia del AG para cualquier función estática. Sin embargo, predecir el comportamiento en el caso de una función arbitraria es más difícil debido a las propiedades no lineales que introduce la selección.

De manera general y asumiendo que la similaridad en las cadenas implica similaridad en los valores de aptitud, podemos plantear la siguiente ecuación para el cambio en el promedio de Hamming por generación:

$$h_{t+1} = f(h_t)$$

1. El teorema de los esquemas [26] indica que $f(h_t)$ es lineal:

$$h_{t+1} = ah_t - b$$

2. En la ausencia de mutación el promedio de Hamming final es cero, lo cual implica que $b = 0$.

Lo anterior nos da la relación:

$$h_{t+1} = ah_t$$

Resolviendo por recurrencia obtenemos la solución general:

$$h_t = \begin{cases} l/2 & t = 0 \\ a^t h_0 & t > 0 \end{cases}$$

De esta manera, en cualquier aplicación específica es posible estimar el valor de a midiendo el promedio de Hamming durante una corrida del AG.

4.2. Modelo Basado en Cadenas de Markov

En el Capítulo 2 se presentaron algunos resultados acerca de la matriz fundamental correspondiente a la matriz de transición de una cadena de Markov. Como vimos, tal matriz puede usarse para calcular los tiempos esperados de convergencia de la cadena. En este capítulo aplicaremos tales resultados a la correspondiente matriz de transición del AGE.

Rudolph [41] mostró que la matriz del AGE tiene la forma:

$$\mathbf{P}^+ = \begin{pmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{R} & \mathbf{T} \end{pmatrix}$$

donde $\mathbf{P}$ es la matriz de transición del AGS y:

$$\mathbf{R} = \begin{pmatrix} \mathbf{PE_{21}} \\ \vdots \\ \mathbf{PE_{2^l,1}} \end{pmatrix} \mathbf{T} = \begin{pmatrix} \mathbf{PE_{22}} & & \\ \vdots & \ddots & \\ \mathbf{PE_{2^l,2}} & \cdots & \mathbf{PE_{2^l,2^l}} \end{pmatrix}$$

donde $\mathbf{E_{ij}}$ son los correspondientes bloques de la matriz de elitismo $\mathbf{E}$.

Puesto que en el modelo de Rudolph la matriz $\mathbf{P}$ corresponde con las poblaciones cuyo súper individuo es el óptimo global, podemos considerar que al estar la cadena en uno de esos estados, se ha terminado el proceso. Luego, cualquier cambio en la población puede ignorarse pues el súper individuo no se modificará. Así pues, podemos reescribir la matriz como:

$$\mathbf{P}^{+} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{R} & \mathbf{T} \end{pmatrix}$$

Podemos ver claramente ahora que la cadena de Markov correspondiente al AGE es absorbente. De acuerdo a la Definición 2.2.4, la matriz fundamental que nos interesa en este caso es:

$$\mathbf{N} = (I - \mathbf{T})^{-1}$$

4.2.1. Estudio de la Matriz del AGE ($\mathbf{P}^{+}$)

Puesto que nuestro objetivo es conocer la matriz fundamental $\mathbf{N}$, en primer lugar procederemos a estudiar la estructura de la matriz bloque $\mathbf{T}$.

Como recordaremos, una vez teniendo la matriz $\mathbf{P}$ se procedió a construir la matriz $\mathbf{P}^{+}$ de la siguiente manera:

$$\mathbf{P}^{+} = \begin{pmatrix} \mathbf{P} & & & \\ & \mathbf{P} & & \\ & & \ddots & \\ & & & \mathbf{P} \end{pmatrix} \begin{pmatrix} \mathbf{E_{11}} & & & \\ \mathbf{E_{21}} & \mathbf{E_{22}} & & \\ \vdots & \vdots & \ddots & \\ \mathbf{E_{2^l,1}} & \mathbf{E_{2^l,2}} & \cdots & \mathbf{E_{2^l,2^l}} \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{PE_{11}} & & & \\ \mathbf{PE_{21}} & \mathbf{PE_{22}} & & \\ \vdots & \vdots & \ddots & \\ \mathbf{PE_{2^l,1}} & \mathbf{PE_{2^l,2}} & \cdots & \mathbf{PE_{2^l,2^l}} \end{pmatrix}$$

Matriz P

En esta sección estudiaremos los elementos de la matriz $\mathbf{P}$. Como sabemos, dicha matriz es resultado del producto:

$$\mathbf{P} = \mathbf{CMS}$$

por lo que a continuación se especificarán los elementos de las matrices correspondientes.

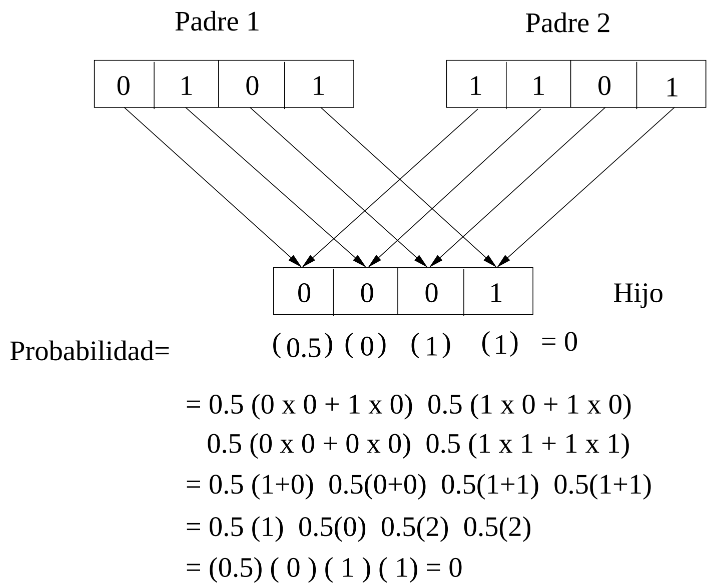

Figura 4.1: Ejemplo del cálculo de la probabilidad de cruza uniforme. El símbolo $\oplus$ se representó con la letra **x**.

Elementos de la Matriz de Cruza

Se modelaron los elementos de 2 tipos de cruza: uniforme y de un punto. Para ello se definió la siguiente operación:

$\oplus$	0	1
0	1	0
1	0	1

La operación $\oplus$ no es más que la negación del $o-exclusivo$, y se aplicará entre los bits correspondientes a una posición fija de un padre y un posible hijo. De manera que si los bits son iguales, el resultado es 1 y es 0 en caso contrario.

I Cruza uniforme:

Cuando se lleva a cabo la cruza uniforme (suponiendo un porcentaje de cruza de 0.5) cada bit de un nuevo hijo tiene 0.5 de probabilidad de ser igual al correspondiente bit de cada uno de los dos padres. Usando este hecho se desarrolló la fórmula siguiente:

$$c_{ij} = \prod_{q=1}^{n} (0.5)^l \prod_{r=1}^{l} \sum_{s=\phi(q)-1}^{\phi(q)} \pi_s^r(i) \oplus \pi_q^r(j)$$

donde $\phi(q) = 2\lfloor \frac{q+1}{2} \rfloor$.

Intuitivamente, dados dos padres y un posible hijo, la probabilidad de obtener el segundo de los primeros no es más que el producto de la probabilidad de obtener de manera independiente cada bit del hijo a partir de los padres. Por otra parte, la probabilidad de obtener un cierto bit del hijo es: 1 si los correspondientes bits de los padres son iguales entre ellos y al bit que se quiere obtener, 0.5 si los bits de los padres son diferentes y 0 si los bits de los padres son iguales entre ellos pero distintos al bit del hijo. La figura 4.1 muestra un ejemplo del cálculo de la probabilidad.

II Cruza de un punto. Asumiendo que el punto de cruza se escoge aleatoriamente se obtiene que:

$$c_{ij} = \prod_{r=1}^{n} \left[\sum_{k=1}^{l} \frac{1}{l} \left(\prod_{s=1+\alpha}^{(k-1)+\beta} (\pi_{\phi(r)}^s(i) \oplus \pi_r^s(q)) \prod_{s=k-\alpha}^{(l-1)-\beta} (\pi_{\phi(r)+1}^s(i) \oplus \pi_r^s(q)) \right) \right]$$

donde $\alpha = (1 - r\text{mod}2)(k-1)$ y $\beta = (1 - r\text{mod}2)(l-k)$.

En este caso, si suponemos fijo el punto de cruza, sólo tenemos que verificar si los bits del primer padre desde el inicio de la cadena hasta el punto de cruza son iguales todos a los correspondientes del hijo y verificar que los que restan sean iguales (todos también) a los correspondientes del segundo padre. En este caso, los términos de la matriz sólo pueden ser cero o uno. De manera que el término total es el resultado de la suma de las probabilidades para cada posible punto de cruza multiplicadas por la probabilidad de elegir cada punto.

Elementos de la Matriz de Mutación

Como recordaremos, Rudolph modela el AG usando mutación uniforme. Los elementos correspondientes son:

$$m_{ij} = p_m^{H_{ij}} (1 - p_m)^{N - H_{ij}}$$

donde p_m es la probabilidad de mutación y H_{ij} es la distancia de Hamming entre las poblaciones i y j. Puesto que la mutación se aplica con probabilidad p_m a cada bit, la expresión del elemento m_{ij} es bastante sencilla.

Elementos de la Matriz de Selección

La selección modelada es la conocida como selección proporcional. Como sabemos, en tal selección cada individuo tiene una probabilidad de ser seleccionado proporcional a su aptitud, de donde:

$$s_{ij} = \begin{cases} \frac{\prod_{k=1}^{n} f(\pi_k(j))}{(\sum_{k=1}^{n} f(\pi_k(i)))^n} & \text{si } \pi_k(j) \in \{\pi_r(i) | r = 1, \cdots, n\} \ \forall k = 1, \cdots, n. \\ 0 & \text{de otra manera.} \end{cases}$$

en donde f es la función objetivo (aptitud).

Elementos de la Matriz **P**

Dado que la matriz **P** es el producto de las matrices de cruza, mutación y selección, tenemos que:

$$p_{ij} = \sum_{p=1}^{n} (\sum_{q=1}^{n} c_{iq} m_{qp}) s_{pj}$$

por lo que:

$$p_{ij} = \sum_{p=1}^{n} \left[\sum_{q=1}^{n} \left(\prod_{r=1}^{n} (0.5)^l \prod_{s=1}^{l} \sum_{t=\phi(r)-1}^{\phi(r)} \pi_t^s(i) \oplus \pi_r^s(q) \right) \left(p_m^{H_{qp}} (1 - p_m)^{N - H_{qp}} \right) \right] \frac{\prod_{k=1}^{n} f(\pi_k(j))}{(\sum_{k=1}^{n} f(\pi_k(p)))^n}$$

Matriz E

La matriz **E** es la siguiente:

$$\begin{pmatrix} \mathbf{E_{11}} & & & \\ \mathbf{E_{21}} & \mathbf{E_{22}} & & \\ \vdots & \vdots & \ddots & \\ \mathbf{E_{2^l,1}} & \mathbf{E_{2^l,2}} & \cdots & \mathbf{E_{2^l,2^l}} \end{pmatrix}$$

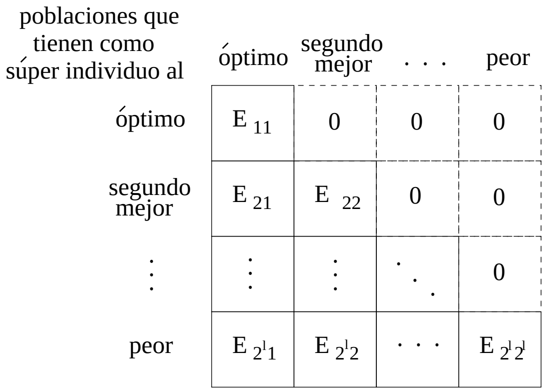

Figura 4.2: La estructura de la matriz de elitismo se debe a que el súper individuo de una población dada sólo puede mejorar, por lo que existen bloques de ceros.

En [41] Rudolph menciona que esta matriz tiene exactamente un 1 por fila además de que sólo $\mathbf{E_{11}}$ es una matriz identidad y las matrices $\mathbf{E_{aa}}$ ($a \geq 2$) son matrices identidad con algunos ceros en la diagonal. A continuación trataremos de aclarar estas aseveraciones y concretar un ejemplo específico de dicha matriz.

Como sabemos:

$$e_{ij} = \begin{cases} 1 & \text{si } f(\pi_0(i)) < f(b) \\ 0 & \text{de otra manera.} \end{cases}$$

donde $b = $argmax$\{f(\pi_k(i))|k = 1, ..., n\} \in \mathbb{B}^l$ y

$$j \stackrel{\text{def}}{=} (b, \pi_1(i), \pi_2(i), ..., \pi_n(i)) \in \mathcal{S}$$

Así pues, dada una población fija es claro que su súper individuo no puede más que mejorar (figura 4.2), de manera que sólo tiene dos opciones:

1. Mantenerse intacta:

$$f(\pi_0(i)) \geq f(b) \Rightarrow e_{ii} = 1$$

2. Actualizar su súper individuo por el máximo de su población:

$$f(\pi_0(i)) < f(b) \Rightarrow e_{ij} = 1$$

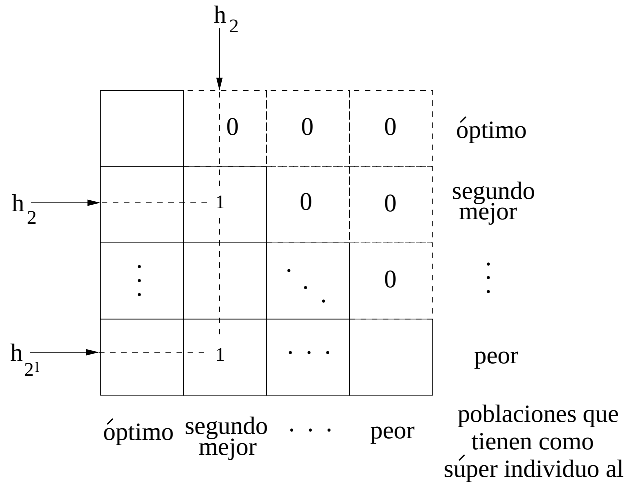

Figura 4.3: Si las poblaciones h_2 y h_{2^l} tienen como máximo al segundo mejor individuo, después de aplicarles la matriz de elitismo ambas pasarán a ser h_2. De esta manera, la matriz de elitismo puede tener más de un 1 en las columnas diferentes de cero.

De lo anterior, concluímos que existe un 1 y sólo un 1 en las filas de la matriz $\mathbf{E}$ y los demás elementos son ceros.

Ahora procederemos a estudiar la estructura de las columnas. Es claro que ninguna población i pasará a ser aquella población j en la que el súper individuo sea de menor calidad que el máximo dentro de ella, es decir, $f(\pi_0(j)) < f(b)$. De aquí se sigue que la columna correspondiente a dicha población j constará completamente de elementos 0, es decir $e_{ij} = 0$ para todo i. Así pues, la matriz $\mathbf{E}$ tiene columnas de ceros.

Procederemos ahora a estudiar aquellas columnas distintas de cero. Recordemos que en la estructura final de la matriz $\mathbf{E}$ cada fila de bloques i representa las probabilidades de transición de todas las posibles poblaciones pero con el $i-\acute{e}simo$ mejor individuo como súper individuo (figura 4.2). Es claro que en cada fila de bloques existe una población h que contiene como máximo al segundo mejor individuo. Consideremos a esa población en el bloque de filas 2 y en el 2^l, y llamémoslas h_2 y h_{2^l}. Claramente, la población h_2 quedará intacta, pero la población h_{2^l} pasará a ser h_2 (figura 4.3). En conclusión, tenemos que es posible encontrar más de un 1 en las columnas distintas de cero.

Consideraremos ahora las columnas pero de bloques. Para la primera columna de bloques (correspondiente a las poblaciones que tienen al óptimo como súper individuo) es claro que en una columna correspondiente a una población cuyo máximo sea el óptimo habrá un 1 en cada bloque, pues esto quiere decir que el súper individuo de esa población pasará a ser el óptimo en caso de no serlo, para cada uno de los bloques. Digamos entonces que todas las poblaciones cuyo máximo es el óptimo se “quedan” en el primer bloque de columnas. Análogamente, las poblaciones cuyo máximo sea el $i - \acute{e}simo$ mejor individuo se “quedarán” en el $i - \acute{e}simo$ bloque. Así pues, en el bloque de columnas i habrá un máximo de $(2^l - i + 1)$ unos por columna. Es decir, tantos unos como bloques haya en la columna puesto que en cada bloque habrá una población cuyo máximo sea el individuo i.

De esta manera, conforme descendemos a lo largo de las filas de bloques, las poblaciones se van “repartiendo” a lo largo de la fila dependiendo de la calidad de su máximo individuo.

Para ejemplificar lo anterior, a partir de este momento asumiremos que dentro de cada bloque las poblaciones se encuentran ordenadas (por conjuntos dentro de los cuales no importa el orden) de acuerdo a la calidad de su máximo individuo. Digamos que se tienen 3 poblaciones y tres posibles súper individuos, la matriz de elitismo correspondiente tendrá la forma que se muestra en la figura 4.4.

Obviamente, el caso que se ha usado como ejemplo resulta bastante irreal pues de hecho sería muy difícil encontrarse con un problema con 3 posibles poblaciones y 3 posibles súper individuos, además de que a cada población se le está asignando un individuo máximo diferente.

En general, es claro que el número de poblaciones cuyo máximo es el $i - \acute{e}simo$ mejor individuo desciende conforme i aumenta (si el máximo individuo es el i, en el resto de la población sólo podemos encontrar $N - i$ individuos distintos); sin embargo, no llega a ser cero para ningún valor de i.

A continuación se presentará el mínimo caso que se apega a las condiciones comunes de los Algoritmos Genéticos.

Consideremos pues el caso más simple. Sea $l = 2$ y $n = 2$, esto nos da un total de $2^{2\times 2} = 2^4 = 16$ poblaciones a considerar por el AG. Sin embargo, sólo con fines de ilustrar más fácilmente nuestro ejemplo consideraremos el hecho de que de manera formal el número neto de poblaciones es [39]:

$$N = \begin{pmatrix} n + r - 1 \\ r - 1 \end{pmatrix}$$

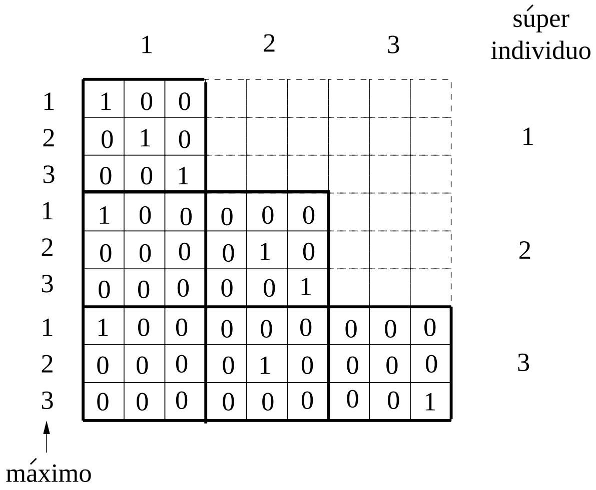

Figura 4.4: En la matriz de elitismo las poblaciones se van "repartiendo" a lo largo de la fila de bloques de acuerdo a la calidad de su individuo máximo conforme se desciende a lo largo de la matriz. La figura muestra el caso en el que se tienen 3 posibles poblaciones y 3 posibles individuos máximos.

donde $r = 2^l$. Por lo tanto en este caso tenemos $N = 10$ posibles poblaciones.

Como existen sólo 4 ($2^l = 4$) posibles individuos diremos que 4 poblaciones tienen como máximo al individuo óptimo, 3 tienen como máximo al segundo mejor, 2 al tercero mejor y 1 al peor individuo. Lo anterior es con el fin de modelar de alguna manera el hecho de que el número de posibles poblaciones que tienen a un cierto individuo como máximo desciende conforme lo hace la calidad de éste.

La correspondiente matriz de elitismo se muestra en la figura 4.5. Se trata de una matriz de 40×40. En ella nuevamente podemos observar cómo las poblaciones se van "repartiendo" a lo largo de las filas de bloques de acuerdo a la calidad de su máximo individuo conforme se desciende a lo largo de la matriz. Las dimensiones de esta matriz nos dan una idea de la complejidad del modelado de un caso con parámetros mayores.

Dado lo anterior, ahora conocemos la estructura de la matriz $\mathbf{P}^+$. La figura 4.6 muestra la matriz $\mathbf{P}^+$ para el caso de 3 poblaciones y 3 individuos discutido anteriormente.

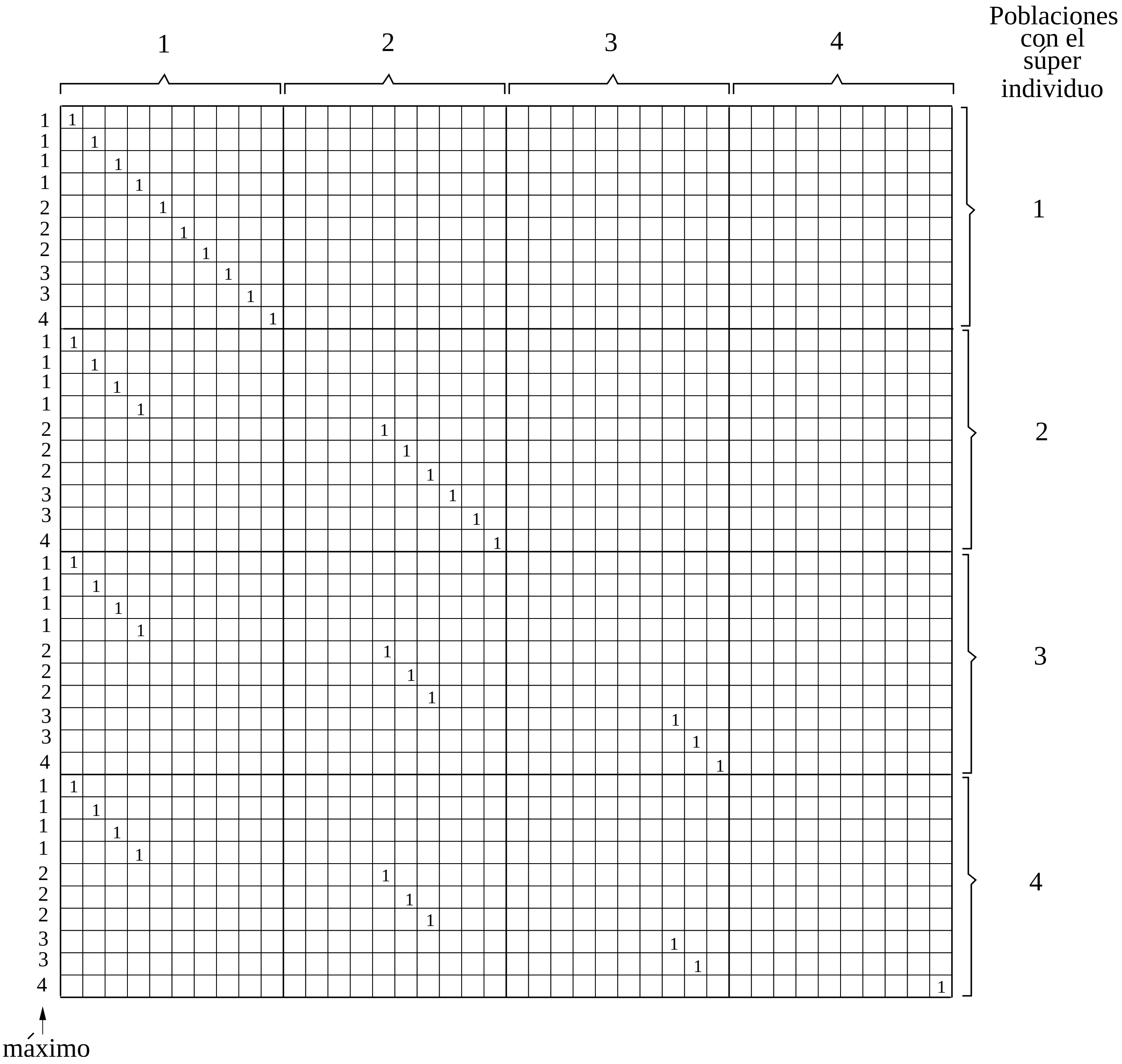

Figura 4.5: Matriz de elitismo para el caso de 4 posibles individuos y 10 posibles poblaciones.

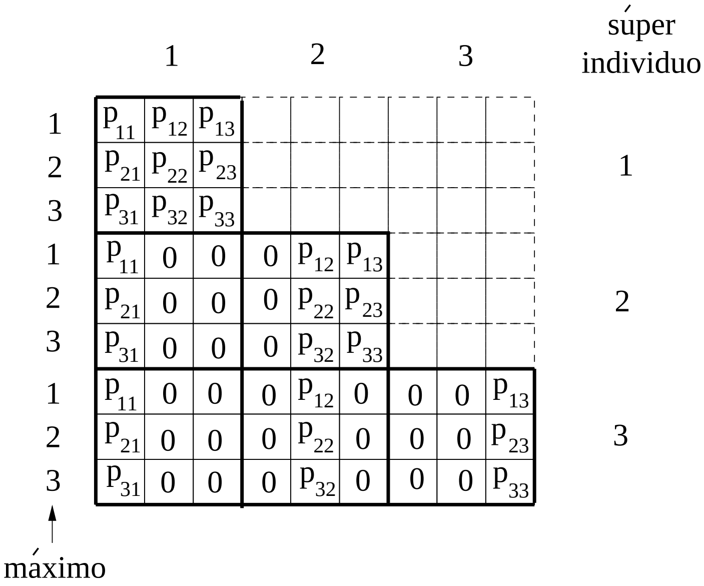

Figura 4.6: $\mathbf{P}^+$ en el caso de 3 posibles poblaciones y 3 posibles individuos.

4.2.2. Tiempo Esperado de Convergencia

Nuevamente usaremos nuestros dos ejemplos anteriores para ilustrar el procedimiento a seguir.

En el caso de 3 individuos y 3 poblaciones, cuya matriz $\mathbf{P}^+$ se muestra en la figura 4.6, el bloque $\mathbf{T}$ correspondiente se muestra en la figura 4.7.

Se usó el paquete MATHEMATICA 4.0 para llevar a cabo los cálculos correspondientes a la matriz fundamental $\mathbf{N}$, en donde

$$\mathbf{N} = (I - \mathbf{T})^{-1}$$

De acuerdo al Teorema 2.2.7, una vez teniendo la matriz $\mathbf{N}$, ésta debe ser multiplicada por un vector columna de unos y el resultado es un vector cuyos elementos son los tiempos esperados de absorción o convergencia para cada uno de los estados transitorios, en términos de los elementos de la matriz $\mathbf{P}$.

A continuación se presentan los resultados obtenidos:

Caso I. 3 individuos - 3 poblaciones. Se tienen 6 estados transitorios:

(i) $\frac{1+p_{12}+p_{13}-p_{22}-p_{13}p_{22}+p_{12}p_{23}+p_{13}p_{32}-p_{23}p_{32}-p_{33}+p_{12}p_{33}+p_{22}p_{33}}{1-p_{22}-p_{23}p_{32}-p_{33}+p_{22}p_{33}}$

(ii) $\frac{-1-p_{23}+p_{33}}{1-p_{22}-p_{23}p_{32}-p_{33}+p_{22}p_{33}}$

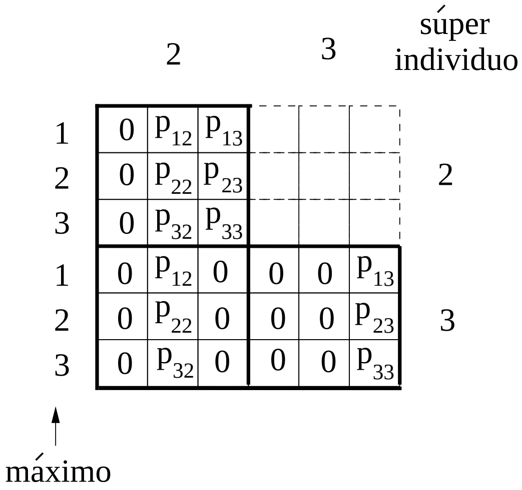

Figura 4.7: El bloque $\mathbf{T}$ en el caso de 3 posibles poblaciones y 3 posibles individuos.

(iii) $\frac{1-p_{22}+p_{32}}{1-p_{22}-p_{23}p_{32}-p_{33}+p_{22}p_{33}}$

(iv) $\frac{1+p_{12}+p_{13}-p_{22}-p_{13}p_{22}+p_{12}p_{23}+p_{13}p_{32}-p_{23}p_{32}-p_{33}+p_{12}p_{33}+p_{22}p_{33}}{1-p_{22}-p_{23}p_{32}-p_{33}+p_{22}p_{33}}$

(v) $\frac{1+p_{23}-p_{33}}{1-p_{22}-p_{23}p_{32}-p_{33}+p_{22}p_{33}}$

(vi) $\frac{-1+p_{22}-p_{32}}{1-p_{22}-p_{23}p_{32}-p_{33}+p_{22}p_{33}}$

Caso II. 4 individuos - 10 poblaciones. Se tienen 30 estados transitorios. El software utilizado no pudo desplegar los resultados. La causa obvia es principalmente que en la expresión de la matriz cuya inversa desea calcularse aparecen 60 variables.

Es claro que el uso de los elementos de la matriz $\mathbf{P}$ como tales, es decir, el manejar cada p_{ij}, es bastante complejo.

Por tal motivo se pensó en la posibilidad de usar una etiqueta p_i por cada columna i de la matriz $\mathbf{P}$ a fin de facilitar los cálculos necesarios. Tales etiquetas p_i deben cumplir: $\sum_{i=1}^{N} p_i = 1$.

Asumamos que podemos etiquetar cada columna i de la matriz $\mathbf{P}$ por el término p_i.

Para fijar ideas, la matriz $\mathbf{P}^+$ y la matriz bloque $\mathbf{T}$ correspondientes al caso de 3 posibles individuos y 3 posibles poblaciones usando las etiquetas p_i se muestran en las figuras 4.8 y 4.9.

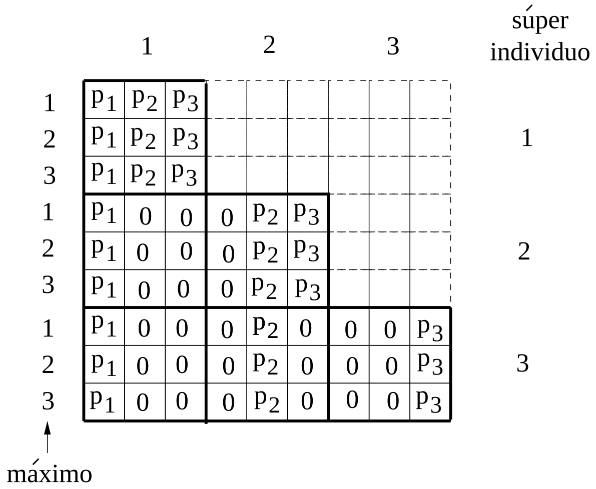

Figura 4.8: Para el caso de 3 poblaciones y 3 individuos, ésta es la matriz $\mathbf{P}^{+}$ en el caso en el que se usa una cota por columna.

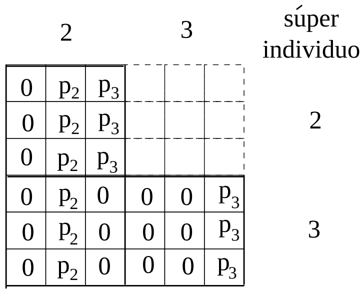

Figura 4.9: Matriz bloque $\mathbf{T}$ correspondiente al caso de la figura 4.8.

A continuación se muestran los resultados obtenidos usando las nuevas matrices:

Caso I. 3 individuos - 3 poblaciones. Se tienen 6 estados transitorios. Tiempos esperados usando una etiqueta para cada columna:

$$\frac{-1}{-1 + p_2 + p_3}$$

para todos los estados transitorios.

Caso II. 4 individuos - 10 poblaciones. Se tienen 30 estados transitorios. Tiempos esperados usando una etiqueta para cada columna:

$$\frac{-1}{-1 + p_5 + p_6 + p_7 + p_8 + p_9 + p_{10}}$$

para todos los estados transitorios.

4.2.3. Experimentos

Se llevaron a cabo las pruebas correspondientes al caso de 10 poblaciones y 4 individuos. En este caso cada población consta de 2 individuos y cada individuo es de longitud 2. Como mencionamos anteriormente, el número neto de poblaciones para el AG en este caso es de 16.

Con fines de poder jerarquizar a los cuatro individuos, se definió la función:

$$f(x_1 x_2) = x_1 + 0.5x_2 + 0.5$$

Por ejemplo:

$$f(01) = 0 + 0.5 * 1 + 0.5 = 1.0$$

Como la función f es estrictamente positiva para cualquier individuo, ésta misma se usó como función de aptitud. En la siguiente tabla se muestran los cuatro posibles individuos, su correspondiente aptitud y su jerarquía:

individuo	aptitud	jeraquía
00	0.5	4
01	1.0	3
10	1.5	2
11	2.0	1

A continuación se presenta la tabla con las correspondientes 16 poblaciones organizadas de acuerdo al individuo máximo correspondiente:

Máximo	No.	*individuos*	
1	1	11	00
1	2	11	01
1	3	11	10
1	4	11	11
1	5	10	11
1	6	01	11
1	7	00	11
2	8	10	00
2	9	10	01
2	10	10	10
2	11	01	10
2	12	00	10
3	13	01	00
3	14	01	01
3	15	00	01
4	16	00	00

Como podemos ver, se tienen 7 poblaciones que tienen como máximo al óptimo, 5 que tienen como máximo al segundo mejor, 3 al tercero mejor y una al peor individuo. Además, dentro de estos grupos no importa el orden.

De acuerdo al modelo descrito en el Capítulo 3, la cadena correspondiente al AGE tiene en total 16 estados absorbentes (u óptimos), es decir, todas las posibles poblaciones pero con el óptimo como súper-individuo, y 48 estados transitorios, o sea, las dieciséis posibles poblaciones pero con el segundo, tercero o peor individuo como súper-individuo.

Usando las expresiones obtenidas y el procedimiento descrito en la sección anterior, se construyeron las correspondientes matrices de cruza, mutación y selección, y posteriormente la correspondiente matriz $\mathbf{P}$. Asimismo, se construyó la matriz $\mathbf{E}$ y finalmente se obtuvo la matriz $\mathbf{P}^+$. A partir de ésta última, se obtuvo la matriz bloque $\mathbf{T}$, la cual da lugar a la matriz fundamental $\mathbf{N}$.

Puesto que en nuestro modelo la probabilidad de cruza (p_c) se fijó con el valor 1, dado que el tamaño de la población y la longitud de los individuos están fijos también, los resultados sólo dependen de la probabilidad de mutación (p_m).

Por otra parte, se corrió un AG con las condiciones impuestas por el modelo dentro de las que se hace notar que el súper-individuo (individuo elitista) no debe intervenir en el proceso evolutivo. Los parámetros fijos para el AG fueron:

$$\begin{aligned} \text{tamaño de población} &= 2 \\ \text{longitud del cromosoma} &= 2 \\ \text{probabilidad de cruza} &= 1.0 \end{aligned}$$

Sea g la variable aleatoria cuyo valor es el número de iteraciones necesarias para la convergencia del AG.

A continuación se muestra el valor esperado de la variable g ($\mathsf{E}[g]$) y la desviación estándar ($\mathsf{D}[g]$) correspondiente, obtenidos (usando MATHEMATICA 4.0) por el modelo teórico (MT) desarrollado y los resultados obtenidos por el AG, para los cuales se hicieron 100 corridas con semilla de aleatorios distinta:

p_m	AG		MT	
	$\mathsf{E}[g]$	$\mathsf{D}[g]$	$\mathsf{E}[g]$	$\mathsf{D}[g]$
0.001	345.05	540.001	514.137	660.781
0.005	90.98	131.4615	103.782	132.47
0.01	40.89	54.01943	52.4913	66.4335
0.03	13.0	16.8367	18.3079	22.4155
0.07	5.08	7.036241	8.56138	9.85261
0.1	4.69	6.5	6.38044	7.03
0.2	2.69	3.47	3.87256	3.78054
0.5	1.32	1.847083	2.52747	1.96485

En la figura 4.10 se muestra la gráfica de los valores obtenidos por ambos métodos. Como podemos ver, los valores obtenidos por el modelo teórico resultan ser mayores que los valores obtenidos por el AG, en todos los casos.

Tal vez los valores que se obtuvieron en los casos de p_m =0.001, 0.005, 0.01 parecerán muy altos, pero son correctos desde el siguiente punto de vista: si recordamos la tabla de todas las posibles poblaciones, aproximadamente el 46 % (poblaciones 1 a 7) contiene al óptimo, y sólo las poblaciones 9 y 11 contienen los esquemas necesarios para dar lugar al óptimo mediante la cruza. En conclusión, aproximadamente el 40 % de las posibles poblaciones no contiene al óptimo ni a los esquemas necesarios para dar lugar a él. Dado que la población es muy pequeña el número de mutaciones llevadas a cabo en

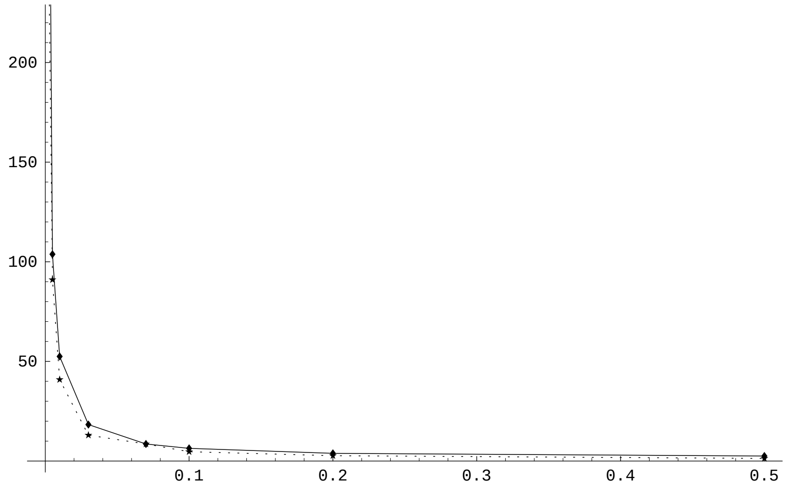

Figura 4.10: Gráfica de los valores obtenidos por el AG y por el modelo teórico desarrollado para la variable g, en función de la probabilidad de mutación. La línea continua corresponde con los valores generados por el modelo teórico y la línea punteada con los valores generados por el AG.

un número corto de generaciones es prácticamente cero, siendo éste el único mecanismo para poder converger correctamente, por lo que se necesita de un número alto de generaciones para encontrar finalmente el óptimo.

Más precisamente, si en cada generación se tienen 4 oportunidades para llevar a cabo una mutación y la probabilidad de mutación es de 0.001, tenemos entonces que se necesitan cuando menos 250 iteraciones en promedio para asegurar que se llevó a cabo al menos una mutación. De ahí que para ciertas poblaciones tome muchas iteraciones el encontrar el óptimo.

En la siguiente tabla se muestran las poblaciones para las cuales es particularmente difícil alcanzar el óptimo. Recuérdese que el súper-individuo no participa en el proceso evolutivo.

Máximo	No.	*individuos*	
2	8	10	00
2	10	10	10
2	12	00	10
3	13	01	00
3	14	01	01
3	15	00	01
4	16	00	00

Por supuesto, el problema discutido es menor conforme se aumenta la probabilidad de mutación, lo cual se refleja en los resultados que se mostraron anteriormente.

Por otra parte, la matriz **P** correspondiente al valor p_m =0.5 fue la única que tuvo la característica de tener una etiqueta p_i para cada columna i. Tal propiedad fue causada muy probablemente por el hecho de que la probabilidad de pasar mediante la mutación de la población i a la población j es la misma independientemente de i y j. En este caso, la fórmula obtenida anteriormente dio los resultados correctos, como era de esperarse.

4.3. Conclusiones

Hemos visto que los modelos existentes para estimar el tiempo de convergencia del AG son bastante burdos y, por la misma razón, bastante lejanos al comportamiento real de dicho algoritmo.

En este capítulo se desarrolló un mecanismo para estimar el tiempo de convergencia del AG mediante un modelo basado en cadenas de Markov.

De manera general, resulta muy complicado obtener una expresión para el valor buscado en términos de los parámetros del AG, porque es necesario conocer todos y cada uno de los elementos de la matriz de transición estudiada. Por este motivo, las dimensiones de tal matriz complican sobremanera los cálculos necesarios.

Por otra parte, los elementos de dicha matriz son lo suficientemente complejos como para clasificarlos o acotarlos de manera adecuada, por lo que el conocimiento de la matriz es prácticamente una condición necesaria.

Los resultados de los experimentos llevados a cabo para el caso más sencillo considerado *real* nos llevan a concluir que el modelo es correcto. Sin embargo, está claro que en tal caso fue relativamente sencillo obtener la matriz **P** correspondiente, proceso que por lo general será cada vez más complicado conforme el tamaño de la población y de los individuos aumenta.

Supongamos que la población consta de 50 individuos de longitud 5, la matriz **P** correspondiente es de tamaño $2^{250} \times 2^{250}$. En general, el tamaño de la matriz crece de manera exponencial.

Así pues, podemos concluir que, desde un punto de vista práctico, las cadenas de Markov no resultan una herramienta teórica recomendable para este análisis. Sin embargo, existen otras alternativas a las que se podría recurrir, como por ejemplo: la mecánica estadística [40], los enfoques de interpretación geométrica [51] y el modelado en base a una búsqueda aleatoria y acoplamiento variable de los operadores genéticos [52].

Capítulo 5

Algoritmos Evolutivos Multi-Objetivo (AEMO)

En este capítulo se presenta una introducción a los Algoritmos Evolutivos Multi-Objetivo. En primer lugar, se define el problema que se quiere resolver en este caso y, posteriormente, se describen los principales Algoritmos Evolutivos que se han desarrollado para tal fin.

5.1. Introducción

Dada la capacidad natural (debido al uso de la población) de manejar simultáneamente varias posibles soluciones a un cierto problema, los Algoritmos Evolutivos han sido usados para resolver problemas de optimización multiobjetivo del tipo:

$$\text{(MO)} \qquad \min \vec{f}(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), \cdots, f_k(\vec{x}))$$

donde $f_i : \mathbb{R}^n \to \mathbb{R}$ para $i = 1, ..., k$ son las funciones objetivo, $\vec{f} : X \subset \mathbb{R}^n \to \mathbb{R}^k$ es la función multiobjetivos, y $\vec{x} \in X$ es llamado vector de variables de decisión.

Para describir el concepto de optimalidad en el que estamos interesados, usaremos la siguiente relación en $\mathbb{R}^k$: decimos que $\vec{u} \leq \vec{v}$ si $u_i \leq v_i$ para todo $i = 1, \ldots, k$, y que $\vec{u} < \vec{v}$ si $\vec{u} \leq \vec{v}$ pero $\vec{u} \neq \vec{v}$. En este último caso, decimos que $\vec{u}$ **domina** a $\vec{v}$ y en lugar de $\vec{u} < \vec{v}$ escribimos $\vec{u} \prec \vec{v}$.

Resumiendo, en el conjunto $\mathcal{F} = \vec{f}(X)$ podemos definir la siguiente relación:

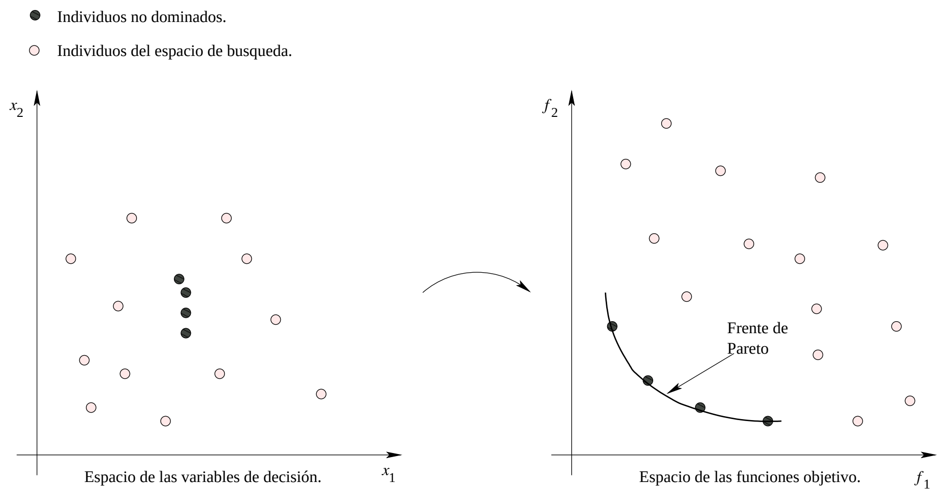

Figura 5.1: Ejemplo de los individuos no dominados y del frente de Pareto para un problema de dos variables y dos objetivos.

Definición 5.1.1 (Dominancia de Pareto) *Un vector $\vec{u} = (u_1, ..., u_k) \in \mathbb{R}^k$ se dice que* **domina** *a $\vec{v} = (v_1, ..., v_k) \in \mathbb{R}^k$ (denotado por $\vec{u} \prec \vec{v}$) si y sólo si $\vec{u} < \vec{v}$.*

Definición 5.1.2 *Un vector de variables de decisión $\vec{x}^* \in X$ es un* **óptimo de Pareto** *para el problema MO si no existe otro vector de variables de decisión $\vec{x} \in X$ tal que $\vec{f}(\vec{x}) \prec \vec{f}(\vec{x}^*)$.*

Sea $X^* = \{\vec{x}^* \in X | \vec{x}^* \text{ es óptimo de Pareto}\}$. Los elementos de X^* son también llamados *no dominados* y el conjunto $\mathcal{F}^* = \vec{f}(X^*)$ se denomina *frente de Pareto*. La figura 5.1 muestra un ejemplo de estos conceptos para un problema en el que tanto el espacio de las variables de decisión como el de los objetivos tienen dimensión dos.

5.2. El Primer AEMO

La primera implementación de lo que ahora se denomina un Algoritmo Evolutivo Multi-Objetivo (AEMO) fue hecha por Schaffer a mediados de los 80's, cuando introdujo el denominado *Vector Evaluated Genetic Algorithm* (VEGA) [46, 47].

VEGA básicamente consiste de un AG simple con un mecanismo de selección modificado. En cada generación, un cierto número de sub-poblaciones son generadas llevando a cabo una selección proporcional de acuerdo a alguna de las funciones objetivo para cada sub-población. Así, para un problema con k funciones objetivo se forman k sub-poblaciones. Posteriormente, las sub-poblaciones vuelven a mezclarse para obtener una población nueva a partir de los operadores genéticos convencionales (cruza y mutación).

Schaffer notó que las soluciones que VEGA generaba eran no dominadas pero sólo respecto a la población actual, lo cual es una deficiencia del algoritmo. Por otra parte, el algoritmo sufre el problema conocido como "especiación", es decir, la formación de especies dentro de la población, formadas por individuos que son muy buenos en diferentes aspectos pero que no necesariamente codifican buenas soluciones compromiso. Este problema se debe claramente al mecanismo de selección utilizado, pues VEGA selecciona individuos que destacan en sólo uno de los objetivos a la vez, dando lugar al principal problema del algoritmo: las posibles soluciones compromiso, es decir, las soluciones buenas en promedio para todos los objetivos, no tienen posibilidades de sobrevivir puesto que no son las mejores en ninguna de las funciones objetivo.

Desde mediados de los 80's hasta mediados de los 90's se desarrollaron algunos AEMO relacionados con algoritmos evolutivos que usaban funciones agregativas (normalmente lineales) [30, 57], ordenamientos lexicográficos [19] y enfoques basados en vectores meta [24].

5.3. AEMO's de Primera Generación

La primera generación de AEMO's surge con la propuesta de David Goldberg en 1989 al analizar VEGA y proponer un mecanismo de selección basado en el concepto de optimalidad de Pareto [23]. Además, Goldberg también sugirió el uso de alguna técnica para mantener diversidad (particularmente los nichos [13]), dado que debido a los ruidos estocásticos, los algoritmos evolutivos tienden a generar una solución única independientemente de la distribución inicial.

Los algoritmos más representativos de esta generación son los siguientes:

1. **Nondominated Sorting Genetic Algorithm** (NSGA)
   Este algoritmo genético fue propuesto por Srinivas y Deb en 1994 [49].

Su principal característica es que antes de llevar a cabo la selección, los individuos son clasificados de la siguiente manera: a todos los individuos no dominados se les asigna jerarquía 1 y una aptitud proporcional al tamaño de la población, que además permita que todos ellos tengan las mismas posibilidades de ser seleccionados. Posteriormente, se les aplica un mecanismo conocido como "*fitness sharing*" (compartición de aptitud) que consiste en determinar una vecindad para cada individuo (en un radio σ_{share} definido por el usuario) y disminuir su aptitud de manera proporcional al número de individuos de su mismo rango que se encuentren dentro de ella. Tales vecindades son llamadas comúnmente nichos.

A continuación, este grupo de individuos de jerarquía 1 es ignorado y el proceso se repite. Esta vez los individuos no dominados tendrán jerarquía 2 y una aptitud menor a la de los individuos de jerarquía 1. El proceso continúa sucesivamente hasta que no queden individuos sin clasificar. La figura 5.2 ilustra el proceso de jerarquización del NSGA.

Puesto que los individuos de jerarquía 1 tienen la aptitud más alta, serán seleccionados un número mayor de veces que el resto de la población, lo que permitirá la búsqueda de individuos no dominados. El mecanismo de compartición de aptitud ayuda a la distribución de la población a lo largo del frente de Pareto.

2. **Niched-Pareto Genetic Algorithm** (NPGA)
   Este algoritmo genético fue propuesto por Horn y Nafpliotis en 1994 [28]. El NPGA usa un esquema de *torneo* basado en dominancia de Pareto: dos individuos de la población son seleccionados de manera aleatoria y son comparados contra un subconjunto de la población también escogido aleatoriamente (normalmente se escoge el 10 % de la población). Si uno de los dos individuos es dominado y el otro no, el individuo no dominado gana el torneo. Si los dos individuos son dominados o no dominados, el ganador se elige de acuerdo a un mecanismo de compartición de aptitud que indica el número de individuos que se encuentran en el mismo nicho de cada contendiente y el individuo con menos vecinos gana. La figura 5.3 ilustra los nichos de dos individuos candidatos. Nótese que el NPGA no asigna jerarquías.

3. **Multi-Objective Genetic Algorithm** (MOGA).
   Propuesto por Fonseca y Fleming en 1993 [18]. En este algoritmo

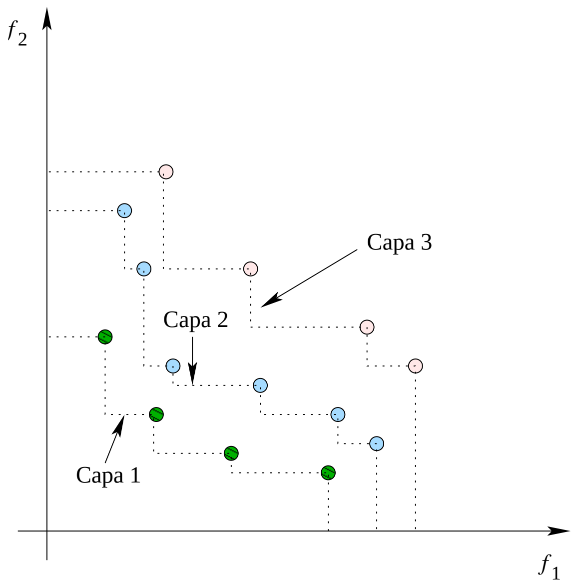

Figura 5.2: Proceso de jerarquización del NSGA.

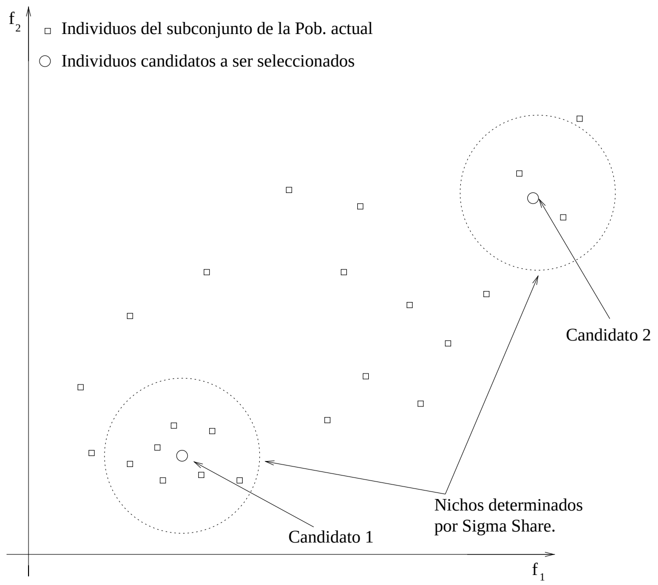

Figura 5.3: Mecanismo de compartición de aptitud aplicado por el NPGA. En este caso el candidato 2 resulta ganador al tener menos vecinos.

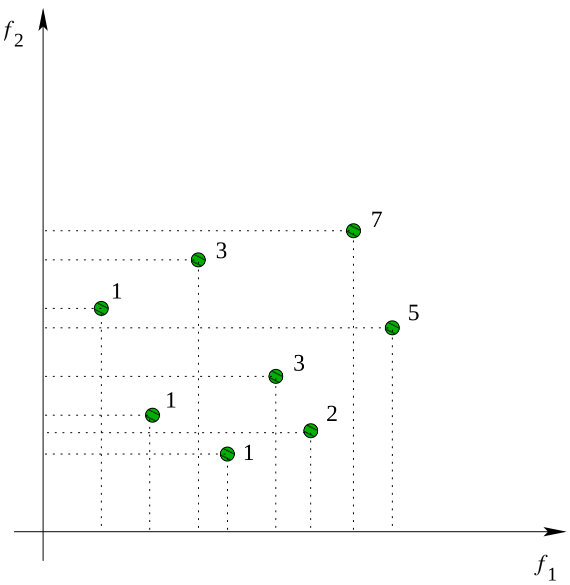

Figura 5.4: Proceso de jerarquización de MOGA. Cada individuo tiene asignada su jerarquía dependiendo del número de individuos por los que es dominado.

genético la jerarquía de cada individuo es proporcional al número de individuos que lo dominan. Por ejemplo, la jerarquía del individuo x_i en la generación t, que es dominado por p_i^t individuos es:

$$jerarquía(x_i, t) = 1 + p_i^t$$

De esta manera, todos los individuos no dominados tienen jerarquía 1 y los demás son castigados de acuerdo al número de individuos que lo dominan. La figura 5.4 ilustra el proceso de jerarquización del MOGA.

La asignación de aptitud se lleva a cabo de la siguiente manera:

- Se ordena la población de acuerdo a las jerarquías.
- Se asigna aptitud interpolando del mejor (jerarquía 1) al peor de acuerdo a alguna función que es usualmente lineal, aunque no necesariamente.
- Se promedian las aptitudes de los individuos de la misma jerarquía.

- Finalmente, se lleva a cabo un mecanismo de compartición de aptitud similar al usado por el NSGA:

  a) Se calcula:

  $$\phi(d_{ij}) = \begin{cases} 1 - (\frac{d_{ij}}{\sigma_{share}}), & d_{ij} < \sigma_{share} \\ 0, & \text{de otra manera} \end{cases}$$

  donde d_{ij} es la distancia euclidiana (en el espacio de las variables o de las funciones objetivo) entre los individuos i y j, y σ_{share} es el radio de los nichos.

  b) La aptitud del individuo i se modifica usando:

  $$f_{S_i} = \frac{f_i}{\sum_{j=1}^{M} \phi(d_{ij})}$$

  donde M es el tamaño de la población.

  Este algoritmo implementa restricciones a la cruza de manera que individuos de nichos distintos no deben cruzarse.

Los AEMO's de la primera generación estuvieron principalmente caracterizados por el uso de mecanismos de selección basados en la dominancia de Pareto y por el uso de diferentes tipos de mecanismos de compartición de aptitud para mantener diversidad.

5.4. AEMO's de Segunda Generación

La introducción de la noción de elitismo dió paso a la segunda generación de AEMO's. El elitismo en optimización evolutiva multi-objetivo normalmente es implementado a través de una población externa, también conocida como población secundaria, en la que se van almacenando los individuos no dominados encontrados a lo largo de la búsqueda. Sin embargo, el elistismo también es posible de implementar mediante el uso de selección $(\mu+\lambda)$, en la que los padres compiten contra los hijos y aquellos no dominados se retienen para la siguiente generación. En cualquiera de los dos casos, eventualmente se hacen necesarias ciertas restricciones que ayuden a obtener un conjunto de individuos no dominados con la mejor distribución posible.

A continuación se presentan los principales algoritmos de esta generación:

1. **Strength Pareto Evolutionary Algorithm** (SPEA)
   Este algoritmo fue propuesto por Zitzler y Thiele [50]. Usa un archivo externo en el que almacena los individuos no dominados que va encontrando, actualizándolo en cada generación. Para cada uno de los individuos que son insertados en el archivo externo, se calcula un valor de *fortaleza* similar a las jerarquías de MOGA puesto que es proporcional al número de individuos que una solución dada domina. La aptitud de cada individuo de la población actual es calculada de acuerdo a los valores de *fortaleza* de los individuos del archivo externo a los cuales domina. Para mantener diversidad, SPEA usa una técnica de *clustering* llamada método de encadenamiento promedio (*average linkage method*) [38].

2. **Strength Pareto Evolutionary Algorithm 2** (SPEA2)
   La segunda versión de SPEA fue propuesta por Zitzler y sus colegas recientemente [58]. El algoritmo tiene básicamente tres diferencias con respecto a SPEA: asigna aptitudes tomando en cuenta la cantidad de individuos que domina y que dominan a una solución dada, usa una técnica de estimación de densidad de vecinos que guía la búsqueda de manera más eficiente e incorpora un mecanismo para truncar el archivo externo que garantiza la preservación de soluciones de frontera.

3. **Pareto Archived Evolution Strategy** (PAES)
   Este algoritmo fue propuesto por Knowles y Corne [33]. Es el algoritmo más simple posible de optimización evolutiva multi-objetivo. Consiste de una estrategia evolutiva (1+1) (un único padre genera un único hijo) en conjunto con un archivo externo en el que se almacenan algunas de las soluciones no dominadas encontradas. Este archivo se usa como una referencia contra cada uno de los individuos que se obtienen como resultado de la mutación.

   La parte más interesante de este algoritmo es la técnica que usa para mantener diversidad: se trata de un rejilla que divide el espacio de búsqueda de manera recursiva y que además es auto-adaptable. Cada individuo es colocado en la celda de la rejilla que le corresponde según sus coordenadas en el espacio de búsqueda, de manera que se mantiene el control del número de individuos que se encuentran en cada celda para obtener una mejor distribución de las soluciones obtenidas.

4. **Nondominated Sorting Genetic Algorithm II** (NSGA-II)

La versión revisada del NSGA fue propuesta recientemente por Deb y sus colegas [12, 14]. El NSGA-II es un algoritmo más eficiente que el NSGA desde un punto de vista computacional debido a que usa un operador de "*crowding*" (agrupamiento) para mantener diversidad. Este operador permite que individuos similares se reemplacen entre sí, esto con el fin de evitar que más y más individuos dominen un mismo nicho. Este proceso es similar al que lleva a cabo PAES mediante su rejilla adaptable. Este algorimo usa elitismo pero no mediante un archivo externo como los algoritmos anteriores sino mediante una selección $(\mu+\lambda)$.

5. **Niched-Pareto Genetic Algorithm 2** (NPGA2)
   Recientemente, Erickson y sus colegas propusieron una versión revisada del NPGA [15]. Esta nueva versión usa jerarquización Pareto pero mantiene la selección de torneo. El elitismo en este algoritmo se incluyó de la misma manera que en el NSGA-II, mediante selección $(\mu + \lambda)$. En este caso, la compartición de aptitud se lleva a cabo contando en cada nicho los individuos de la generación (incompleta) que se está formando y no la actual.

6. **Pareto Enveloped-based Selection Algorithm** (PESA)
   Este algoritmo fue propuesto por Corne y sus colegas [10]. El algoritmo es muy parecido a PAES, pues la población interna es pequeña mientras que la población externa es más grande y además usa la misma rejilla adaptable para mantener diversidad. Sin embargo, ahora la población externa no sólo determina el esquema de diversidad sino que tambien determina el mecanismo de selección del algoritmo decidiendo qué soluciones ingresan en el archivo externo mediante el operador de agrupamiento implícito en la rejilla adaptable.

   Existe una versión revisada de PESA propuesta por Corne y sus colegas recientemente [9] a la que denominó PESA-II. La principal diferencia entre PESA-II y PESA es que en este caso la selección está basada en regiones y no en individuos. Esta modificación se hizo principalmente para mejorar el costo computacional del algoritmo original.

7. **Micro Genetic Algorithm** (micro-GA)
   Este algoritmo fue propuesto por Coello y Toscano [8]. Se trata de un AG con una población muy pequeña (cuatro individuos) y un proce-

so de reinicialización. La figura 5.5 muestra el funcionamiento de este algoritmo.

En primer lugar, se genera un población de manera aleatoria. Esta población entra en la memoria de población, la cual está dividida en dos porciones: una reemplazable y una no reemplazable. La porción no reemplazable no cambia en ningún momento a lo largo de toda la corrida del algoritmo y se desempeña como un medio para proveer la diversidad requerida. Por otro lado, la memoria reemplazable cambia en cada ciclo del micro-GA.

En cada ciclo del micro-GA, la población es tomada (con una cierta probabilidad) de ambas porciones de la memoria, por lo que existe una mezcla de individuos generados aleatoriamente (porción no reemplazable) e individuos "evolucionados" (porción reemplazable). Durante cada ciclo, el micro-GA aplica los operadores genéticos de manera convencional. Después de que termina un ciclo, dos vectores no dominados son escogidos de la población final y son comparados con el contenido de la población externa, la cual está inicialmente vacía. Si uno de ellos (o ambos) siguen siendo no dominados, se incluyen en dicha memoria y se eliminan todos los individuos dominados.

De esta manera, el micro-GA usa tres formas de elitismo: retiene individuos no dominados encontrados en cada ciclo, usa una memoria reemplazable que se actualiza a cada cierto número de intervalos y reemplaza la población por las mejores soluciones obtenidas.

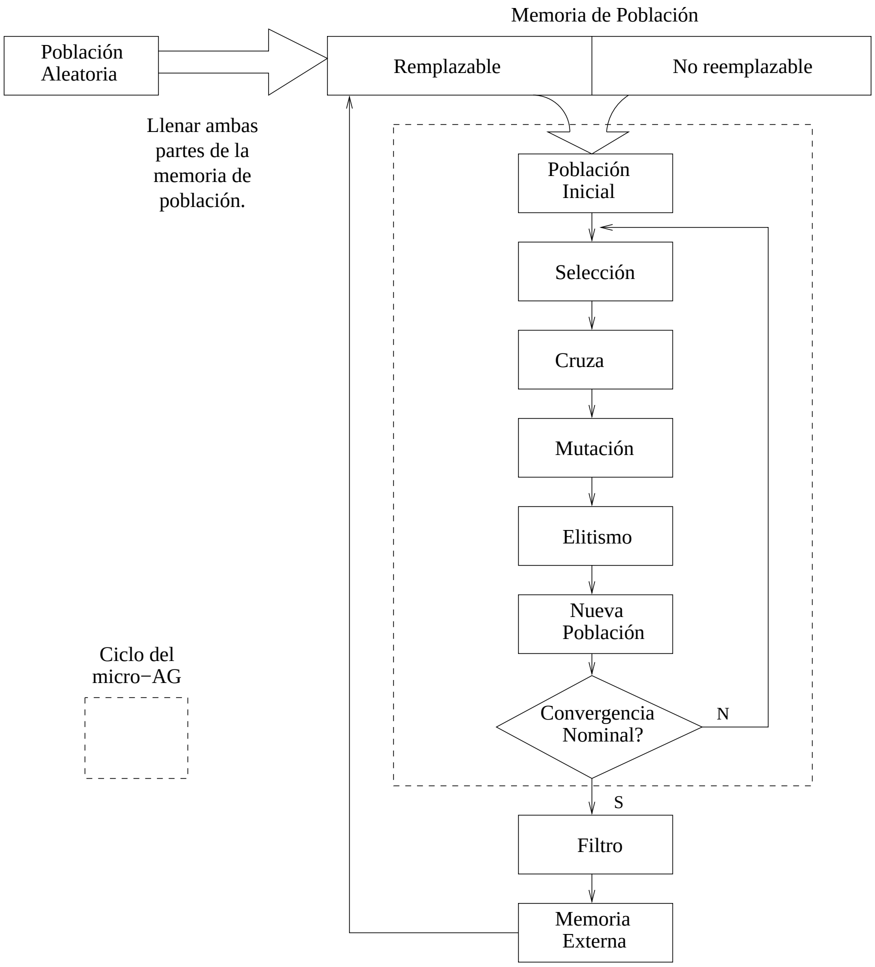

Figura 5.5: Funcionamiento del micro-AG.

Capítulo 6

Teoría de AEMO

El marco teórico correspondiente a la optimización evolutiva multi-objetivo es completamente diferente al que envuelve a la optimización global pues en este último caso el objetivo de la búsqueda es obtener una solución única. Sin embargo, cuando existen varios objetivos por optimizar, la tarea se vuelve mucho más complicada puesto que en general no existe una solución única al problema sino más bien un conjunto de soluciones compromiso.

En este capítulo se presentan y discuten los principales resultados teóricos que se han obtenido hasta la fecha, correspondientes a AEMOs. Por tal razón, a continuación se presentan los preliminares al estudio de los algoritmos encargados de dicha tarea.

6.1. Espacios Parcialmente Ordenados

Si la relación reflexiva, antisimétrica y transitiva $\preceq$ es válida en X entonces el par $(X, \preceq)$ es llamado un conjunto parcialmente ordenado. Si x y y son dos elementos de X tales que $x \preceq y$ pero $x \neq y$, entonces se dice que $x \prec y$.

Definición 6.1.1 *Un elemento $x^* \in X$ es llamado* **elemento minimal** *de $(X, \preceq)$ si no existe $x \in X$ tal que $x \prec x^*$. El conjunto de todos los elementos minimales, denotado por $\mathcal{M}(X, \preceq)$, se dice que es* **completo** *si para cada $x \in X$ existe un $x^* \in \mathcal{M}(X, \preceq)$ tal que $x^* \preceq x$.*

Distintos puntos $x, y \in X$ se dice que son **comparables** *cuando $x \prec y$ o $y \prec x$; de otra manera, x y y son* **incomparables** *lo cual se denota como $x \parallel y$.*

Si cada par de puntos distintos de $(X, \preceq)$ es comparable entonces $(X, \preceq)$ es llamado un conjunto totalmente ordenado o una **cadena**. *Si cada par de puntos distintos de $(X, \preceq)$ es incomparable entonces $(X, \preceq)$ es llamado una* **anticadena**.

Lema 6.1.1 *Si $(X, \preceq)$ es un conjunto parcialmente ordenado y $0 < |X| < \infty$ entonces $\mathcal{M}(X, \preceq)$ es completo.*

Sea ahora $f : \mathbb{B}^l \to \mathcal{F} = \{f(x) : x \in \mathbb{B}^l\} \subset \mathbb{R}^n$. Notemos que la Dominancia de Pareto (definida al inicio del Capítulo 5) define una relación de orden estricto en $\mathcal{F}$.

Por lo tanto:

$$(\mathcal{F}, \preceq) \text{ es un conjunto parcialmente ordenado y}$$
$$\mathcal{M}(\mathcal{F}, \preceq) \simeq \text{frente de Pareto.}$$

Como hemos visto, los objetivos de la búsqueda en este caso son los elementos minimales del espacio de búsqueda.

En secciones posteriores, dado $X \subset \mathbb{B}^l$, se usará eventualmente la notación: $\mathcal{M}_f(X, \preceq) := \mathcal{M}(f(X), \preceq)$.

6.2. Primera Generación

Como recordaremos, los AEMO de primera generación se caracterizaron por el uso de la jerarquización de Pareto y de los mecanismos de tipo *sharing* para mantener diversidad en la población.

Los algoritmos que se describieron en el capítulo anterior correspondientes a esta generación fueron: NSGA, NPGA y MOGA. Como pudimos ver, los tres son AG's y como cualquier algoritmo evolutivo pueden representarse a través de su correspondiente matriz de transición. De esta manera, el modelo presentado en el Capítulo 3 se aplica perfectamente a tales algoritmos.

Si recordamos, en el Capítulo 3 se modeló un AG con matriz de transición irreducible, para lo cual se necesitó que la matriz de transición de la mutación fuera positiva y la de la selección columna permisible, lo cual está asegurado si la mutación usada es uniforme y la selección es proporcional o de torneo. De esta manera, los algoritmos de primera generación mostrados anteriormente pueden adaptarse a tales condiciones.

A continuación propongo una definición de convergencia análoga a la definición 3.1.1, pero para el contexto de optimización multiobjetivo:

Definición 6.2.1 *Sea* $M_t = \#\{\pi_k^t(i) | \pi_k^t(i) \in \mathcal{M}(\mathcal{F}, \preceq)\}$ *una secuencia de variables aleatorias representando el número de elementos minimales del espacio del búsqueda dentro de la población representada por el estado* i *en el paso* t*. Un algoritmo genético multi-objetivo converge al conjunto de elementos minimales del espacio de búsqueda si y sólo si:*

$$lim_{t\to\infty} P\{M_t = tam_{pob}\} = 1$$

donde tam_{pob} *es el tamaño de la población del algoritmo.*

Así pues, la convergencia está en términos del número de elementos de $\mathcal{M}(\mathcal{F}, \preceq)$ que contiene la población del algoritmo en un cierto paso t. De esta manera, cada población será etiquetada con el número de elementos minimales del espacio de búsqueda que contiene.

A continuación presento un teorema en el que se demuestra que tales algoritmos, al igual que cualquier AEMO de primera generación con matriz de transición irreducible (primitiva), no converge en términos de la definición anterior.

Teorema 6.2.1 *Un AEMO de primera generación con matriz de transición primitiva no converge a* $\mathcal{M}(\mathcal{F}, \preceq)$.

Demostración Sea $i \in \mathcal{S}$ cualquier estado en el que $M_t < tam_{pob}$ y $p_i(t)$ la probabilidad de que el AEMO esté en tal estado i en el paso t. Claramente, $P\{M_t \neq tam_{pob}\} \geq p_i(t) \Leftrightarrow P\{M_t = tam_{pob}\} \leq 1 - p_i(t)$. Por el Teorema 2.2.1 la probabilidad de que el AEMO esté en el estado i converge a $p_i(\infty) > 0$. Por lo tanto:

$$lim_{t\to\infty} P\{M_t = tam_{pob}\} \leq 1 - p_i(\infty) < 1$$

y por lo tanto, la condición de convergencia no se satisface. ∎

Una vez demostrado que un AEMO de primera generación con matriz de transición primitiva no puede converger al frente de Pareto en términos de la Definición 6.2.1, queda por estudiar de qué manera afectan en la distribución de la población los mecanismos de compartición de aptitud (*fitness sharing*).

6.2.1. Nichos

Puesto que los estados de la cadena de Markov que modela el algoritmo genético tanto en su versión para un solo objetivo como para multi-objetivo corresponden con la población del algoritmo en sí, y dado que los mecanismos de compartición de aptitud (es decir, la inducción de nichos) introducen modificaciones sólo a los valores de aptitud de cada individuo, resulta complicado modelar los cambios que pueden generar la inducción de nichos en la matriz de transición.

Es claro que la matriz que es modificada por la inducción de nichos es la del operador de selección.

Como un ejemplo de la complejidad del estudio de los cambios en esta matriz, a continuación se presentará el trabajo de Jeffrey Horn [27] en el que se realiza tal estudio usando un modelo bastante simplificado.

Posteriormente se presentará de manera breve el trabajo realizado por Samir W. Mahfoud [36] en el que propone un modelo distinto para los algoritmos que usan nichos y de hecho hace ver que usar cadenas de Markov para modelar tales algoritmos es innecesario en un cierto caso particular.

Modelo de Horn

El modelo de AG usado por Horn fue desarrollado originalmente por Goldberg & Segrest [22] y es bastante sencillo: Consideraremos una población de tamaño N en la que los individuos son cromosomas binarios de longitud 1, es decir, cada individuo puede ser 1 o 0. Es claro que tal población como cadena de Markov tiene un espacio de estados posibles de cardinalidad $(N+1)$ en el que el estado i es aquel en el que se tienen i 1's y $(N-i)$ 0's.

Las probabilidades de transición se calcularán asumiendo el uso de un algoritmo genético generacional con selección proporcional **sin mutación**. De esta manera, la probabilidad de seleccionar al individuo k es:

$$f_k / \sum_i f_i$$

donde f_k denota la aptitud del individuo k. Supongamos que la población se encuentra en el estado i, definiendo la aptitud de un "1" como f_1 y la de un "0" como f_0, la probabilidad de seleccionar un 1 para la siguiente población es:

$$p_1 = \frac{i * f_1}{i * f_1 + (N-i) * f_0} \Rightarrow p_1 = \frac{r * i}{r * i + (N-i)} \qquad (6.2.1)$$

donde r es la razón (o proporción) f_1/f_0. La probabilidad de seleccionar un 0 es:

$$p_0 = \frac{N-i}{i * r + (N-i)}$$

Así, la probabilidad de pasar del estado i al estado j es:

$$p_{ij} = \left(\begin{array}{c} N \\ j \end{array} \right) (p_1)^j (p_0)^{N-j}$$

Sustituyendo los valores de p_1 y p_0:

$$p_{ij} = \left(\begin{array}{c} N \\ j \end{array} \right) \left(\frac{i * r}{i * r + (N-i)} \right)^j \left(\frac{N-i}{i * r + (N-i)} \right)^{N-j}$$

Como podemos ver, las probabilidades p_{ij} son precisamente los elementos de la matriz de selección.

A continuación procederemos a analizar de qué manera se ven afectadas tales probabilidades por la inducción de nichos. Cada pico en el paisaje de aptitud puede ser considerado como un nicho [27], por lo que lo mejor sería que el AG encontrara los mejores nichos y los "llenara" en proporción a su calidad.

Como sabemos, la inducción de nichos modificará la aptitud del individuo h de la manera [21]:

$$f_h/m_h$$

donde $m_h = \sum_{k=1}^{N} sh(d_{hk})$ y $sh(d_{hk}) = 1 - (\frac{d_{hk}}{\sigma_{share}})^{\alpha}$. En este caso d_{hk} es la distancia de Hamming entre los individuos h y k.

Consideraremos dos casos de estudio: (1) nichos que no se traslapan, es decir, nichos aislados, cada uno centrado en un óptimo local (estado perfecto) y (2) nichos que se traslapan.

En el caso de los nichos que no se traslapan tenemos que $\sigma_{share} > 1$ y de esta manera $m_1 = i$ y $m_0 = N - i$ (recordemos que se está usando distancia de Hamming), suponiendo que nos encontramos en el estado i. Por lo tanto las aptitudes son ahora:

$$f_1/i$$

y

$$f_0/(N-i)$$

Por lo tanto, sustituyendo estos valores en la ecuación (6.2.1):

$$p_{ij} = \left(\begin{array}{c} N \\ j \end{array} \right) \left(\frac{r}{r+1} \right)^j \left(\frac{1}{r+1} \right)^{N-j}$$

Es importante hacer notar que estas ecuaciones no son válidas para los valores de $i = 0$ o $i = N$. Estos casos corresponden a los estados absorbentes de la cadena de Markov, aquellos en los que toda la población consta de 1's o de 0's respectivamente, puesto que una vez estando en ellos no es posible obtener alguna variación y pasar a cualquier otro estado distinto (no hay mutación).

En el caso más general, es decir, en el que los nichos se traslapan tenemos que $\sigma_{share} \leq 1$ y por lo tanto: $m_1 = i + (N-i)(1 - 1/\sigma_{share})$ en lugar de i. Análogamente $m_0 = N - i + i(1 - 1/\sigma_{share})$. De esta manera las nuevas probabilidades de transición son (después de muchas operaciones y reacomodos):

$$p_{ij} = \left(\begin{array}{c} N \\ j \end{array} \right) \left(\frac{1}{1 + \frac{(N-i)(N + \frac{i-N}{\sigma_{share}})}{ir(N - \frac{i}{\sigma_{share}})}} \right)^j \left(\frac{1}{1 + \frac{ir(N - \frac{i}{\sigma_{share}})}{(N-i)(N + \frac{i-N}{\sigma_{share}})}} \right)^{N-j}$$

Como podemos ver, las probabilidades de transición correspondientes a la matriz del operador de selección en este caso se vuelven bastante complejas y, como se mencionó, se trata del caso más general y por lo tanto más común.

Ahora recordemos que en nuestro modelo los individuos son de longitud 1 por lo que sólo tenemos dos posibles valores de aptitud y, además, la distancia usada es la de Hamming lo que nuevamente nos da sólo dos posibles valores de distancia. En resumen, el modelo expuesto es lo bastante sencillo como para estimar la complejidad de llevar el estudio análogo para los AGs convencionales.

En [27] Horn muestra gráficamente el comportamiento de la población en cuanto a su distribución y aplica el AG mencionado con diferentes valores de r y de σ_{share} los cuales básicamente se refieren al número de 1's y 0's existentes en la población. Por otra parte, estima también la relación de estos parámetros con el tiempo de absorción de la cadena. Es decir, el tiempo necesario para que se encuentre en un estado absorbente que de alguna manera podría considerarse como convergencia en este sencillo modelo.

Modelo de Mahfoud

En [36] Mahfoud presenta una alternativa al análisis con cadenas de Markov para el estudio de los algoritmos que usan nichos para optimización multimodal. En este trabajo, Mahfoud recalca la importancia de estudiar la distribución esperada-obtenida de los elementos de la población dado el uso de nichos y el tiempo esperado de pérdida de soluciones.

Según Mahfoud, un buen método de nichos debiera poder localizar un cierto número de los mejores b picos de la función, de tal manera que el conjunto de metas son todas aquellas distribuciones de la población que tienen al menos un individuo en cada uno de esos b picos.

La idea principal del modelo de Mahoud consiste en particionar el espacio de búsqueda en distintas *clases de equivalencia*. Ahora bien, se esperaría que cada una de las clases de equivalencia corresponda con un óptimo local del espacio de búsqueda por lo que, puesto que sólo para ciertas funciones cada óptimo local corresponde con alguna partición inducida por determinados esquemas, Mahfoud decide no usar tales particiones para determinar la clase de equivalencia a la que pertenece un cierto individuo.

Así pues, para poder determinar la clase de equivalencia de un individuo, el espacio de búsqueda se particiona con base en el concepto de *atractores*. Un atractor es cada uno de los b mínimos locales y la clase de equivalencia correspondiente a él consta de todos aquellos individuos que se encuentran en su *base de atracción*; se dice que un individuo se encuentra en la base de atracción de un cierto máximo local si y sólo si un algoritmo de *hillclimbing* determinístico (definido de manera especial) lo lleva a tal punto.

Cada clase i tendrá una "aptitud representativa" f_i, la cual se define como la aptitud más alta dentro de ella y todos lo individuos en dicha clase tendrán el mismo valor de aptitud.

El modelo desarrollado por Mahfoud tiene como principal objetivo el estudio de la pérdida de las clases de equivalencia más que de la formación, por lo que se supondrá que todas las clases deseadas se encuentran en la población inicial. **El uso de mutación se suprime**.

Es muy importante mencionar que el modelo presupone que el algoritmo es capaz de determinar siempre y para todo individuo la clase de equivalencia a la que pertenece. Es decir, el modelo asume que los nichos o clases de equivalencia no se traslapan, lo que corresponde con el primer caso estudiado por Horn en el modelo que se presentó en la sección anterior.

Horn calculó las probabilidades de transición correspondientes a este caso y Mahfoud hizo notar que tienen la propiedad de definir una matriz con columnas idénticas. En otras palabras, la población del algoritmo en el tiempo t es **independiente** de la población en el tiempo $t-1$. Por esta razón, se concluye que el análisis de cadenas de Markov resulta una herramienta demasiado compleja e innecesaria en este caso.

En [36] Mahfoud lleva a cabo por separado el estudio de los distintos métodos de compartición de aptitud para inducir nichos de la siguiente manera:

1. *Crowding* & Selección
2. *Crowding* & Selección + Cruza
3. *Sharing* & Selección
4. *Sharing* & Selección + Cruza

A continuación se presentará el caso 3.

La técnica de selección modelada es la de ruleta (selección proporcional), por lo que la probabilidad de seleccionar algún elemento de una clase arbitraria i, $P_s(i)$ es:

$$P_s(i) = \frac{I_i f_i}{\sum_{j=0}^{c-1} I_j f_j}$$

donde I_j es el número de elementos en la clase j. Así, el número esperado de elementos de la clase i en la población después de una generación es:

$$\mu_i = nP_s(i)$$

donde n es el número de individuos en la población.

El valor de aptitud resultado de aplicar el método de compartición de aptitud estará definido por: $f_i' = f_i/I_i$. Por lo tanto:

$$P_s(i) = \frac{I_i f_i'}{\sum_{j=0}^{c-1} I_j f_j'} = \frac{f_i}{\sum_{j=0}^{c-1} f_j}$$

y este caso μ_i se mantiene constante.

Dada la definición de $P_s(i)$, la probabilidad de seleccionar un individuo que no está en la clase i es: $1 - P_s(i)$. De esta manera, después de una generación completa la probabilidad de perder a todos los elementos de la clase i es simplemente la probabilidad de escoger n individuos de otra clase:

$$(1 - P_s(i))^n$$

Supongamos que se tienen c clases, la probabilidad X de que una o más clases desaparezcan después de una generación es:

$$X \leq \sum_{i=0}^{c-1}(1 - P_s(i))^n$$

La probabilidad de mantener a todas las clases a lo largo de una generación es: $1-X$. En la ecuación anterior se tiene una desigualdad porque no han sido restados los términos correspondientes a los eventos intersección, es decir, la probabilidad de que desaparezcan dos o más clases a la vez.

La pérdida de una o más clases puede ser vista como una variable binomial:

- La probabilidad de perder una o más clases después de una generación es: X.
- Después de dos generaciones: $(1 - X)X$.
- Después de g generaciones: $(1 - X)^{g-1}X$.

Sea L el número de generaciones necesarias para perder cuando menos una clase, entonces:

$$P(L = g) = (1 - X)^{g-1}X$$

Por lo tanto, la probabilidad de perder, en g generaciones o menos, todos los elementos de una o más clases es:

$$P(L \leq g) = \sum_{i=1}^{g}(1 - X)^{i-1}X = X\sum_{i=1}^{g-1}(1 - X)^i$$

$$= X\frac{1 - (1 - X)^g}{1 - (1 - X)} = 1 - (1 - X)^g$$

donde $0 < X < 1$ y X es independiente de i.

Así, la probabilidad de mantener a todas las clases por al menos g generaciones es:

$$\gamma = 1 - P(L \leq g) = (1 - X)^g$$

6.3. Segunda Generación

Como se comentó en el Capítulo 5, el elitismo en los AEMO se ha implementado de dos maneras distintas: mediante una población externa o mediante el uso del esquema de selección $\mu + \lambda$.

Existen trabajos teóricos relacionados con ambas opciones de implementar el elistismo. A continuación se presentarán y discutirán tales trabajos.

6.3.1. Elitismo Mediante el Uso de una Población Externa

En [45] Rudolph & Agapie demuestran la convergencia de cuatro algoritmos evolutivos multiobjetivo. Los cuatro algoritmos evolutivos que se estudian en dicho trabajo son algoritmos poblacionales y elitistas, diferenciándose precisamente en esta segunda propiedad pues dos de ellos usan una población externa y los dos restantes un enfoque muy similar al de la selección $\mu + \lambda$.

Los dos algoritmos que usan una población externa se diferencian en lo siguiente:

- **Algoritmo 1.** En la población externa se almacenan los elementos minimales encontrados a lo largo de todo el proceso evolutivo. El tamaño de la población externa puede variar hasta converger al número de elementos minimales que existan en el espacio de búsqueda.

- **Algoritmo 2.** En la población externa se almacenan los elementos minimales encontrados a lo largo de todo el proceso evolutivo pero en este caso dicha población tiene un tamaño límite fijo.

El Algoritmo 1 es el siguiente:

$B(0)$ es generado aleatoriamente a partir de $\mathcal{S} = \mathbb{B}^{nl}$
$A(0) = \mathcal{M}_f(B(0), \preceq)$
$t = 0$
repetir
$B(t+1) = \text{genera}(B(t))$
$A(t+1) = \mathcal{M}_f(A(t) \cup B(t+1), \preceq)$
$t \leftarrow t+1$
hasta Criterio de terminación satisfecho

A fin de poder definir la convergencia para algoritmos evolutivos multiobjetivo, definimos la métrica:

Definición 6.3.1 *Si A y B son subconjuntos de un conjunto finito X entonces:*

$$d(A, B) = |A \cup B| - |A \cap B|$$

es una métrica en 2^X.

De esta manera:

Definición 6.3.2 *Sea $A(t)$ la población de algún algoritmo evolutivo en la iteración $t \geq 0$ y $F_t = f(A(t))$ su conjunto imagen asociado. Se dice que el algoritmo evolutivo converge con probabilidad 1 al conjunto de elementos minimales si:*

$$d(F_t, \mathcal{M}(\mathcal{F}, \preceq)) \to 0 \text{ con probabilidad 1 cuando } t \to \infty.$$

La convergencia del Algoritmo 1 se demuestra con el siguiente teorema:

Teorema 6.3.1 *Si la secuencia $(B(t))_{t \geq 0}$ del Algoritmo 1 es una cadena de Markov finita homogénea con matriz de transición irreducible entonces $d(f(A(t)), \mathcal{M}(\mathcal{F}, \preceq)) \to 0$ con probabilidad 1 cuando $t \to \infty$.*

Demostración Por construcción, el conjunto $f(A(t))$ es una anticadena. Además, tan pronto como un elemento de $\mathcal{M}(\mathcal{F}, \preceq)$ entra en $f(A(t))$ se quedará allí por siempre.

Por lo tanto resta mostrar que todos los elementos de $\mathcal{M}(\mathcal{F}, \preceq)$ estarán contenidos en $f(A(t))$ para algún tiempo aleatorio τ con $\mathbf{P}\{\tau < \infty\} = 1$.

El argumento es muy sencillo:
Supongamos que existe un $a \in A(t)$ tal que $f(a)$ no es un elemento minimal. Puesto que $\mathcal{M}(\mathcal{F}, \preceq)$ es completo existe un $x \in \mathbb{B}^l$ tal que $f(x) \prec f(a)$. El Teorema 2.2.1 asegura que cada elemento de $\mathcal{S}(= \mathbb{B}^{ln})$ será visitado un número infinito de veces. Por lo tanto, el tiempo de espera para la primera ocurrencia de x es finito con probabilidad 1. Esto quiere decir que cada elemento no minimal es eliminado en un número finito de iteraciones.

Además, la irreducibilidad de la cadena de Markov asegura también la aparición en tiempo finito de todos los elementos minimales. En resumen: Todos los elementos minimales entrarán en el conjunto $A(\cdot)$ en tiempo finito y con probabilidad 1. ■

El Algoritmo 2 es el siguiente:

$B(0)$ es generado aleatoriamente a partir de $\mathcal{S} = \mathbb{B}^{nl}$
$A(0) = \mathcal{M}_f(B(0), \preceq)$
$t = 0$
repetir
  $B(t+1) = \text{genera}(B(t))$
  $B^*(t+1) = \mathcal{M}_f(B(t+1), \preceq)$
  $C(t+1) = \emptyset$
  para_cada $b \in B^*(t)$
    $D_b = \{a \in A(t) : f(b) \prec f(a)\}$
    si $D_b \neq \emptyset$ **entonces** $A(t) \leftarrow (A(t) \setminus D_b) \cup \{b\}$
    si $\forall a \in A(t) : f(a) || f(b)$ **entonces** $C(t+1) \leftarrow C(t+1) \cup \{b\}$
  fin de para_cada
  $k = \min\{m - |A(t)|, |C(t+1)|\}$
  $A(t+1) = A(t) \cup \text{ extrae}(k, C(t+1))$
  $t \leftarrow t + 1$
hasta Criterio de terminación satisfecho

Como podemos ver, el Algoritmo 2 tiene como meta, además de almacenar los elementos minimales en la población externa, mantener fijo el tamaño de ésta (constante m), pues la función extrae(k, C) regresa un conjunto de a lo más k elementos distintos del conjunto C tomados a través de un método arbitrario.

A continuación se presenta la demostración de la convergencia al frente de Pareto del Algoritmo 2.

Teorema 6.3.2 *Si la secuencia $(B(t))_{t\geq 0}$ del Algoritmo 2 es una cadena de Markov finita homogénea con matriz de transición irreducible, entonces $d(f(A(t)), \mathcal{M}(\mathcal{F}, \preceq)) \to 0$ y $|A(t)| \to min\{m, |\mathcal{M}(\mathcal{F}, \preceq)|\}$ con probabilidad 1 cuando $t \to \infty$.*

Demostración Por construcción, el conjunto $f(A(t))$ es una anticadena y por lo tanto el conjunto de elementos minimales del conjunto $(f(A(t)), \preceq)$.

Un elemento $a \in A(t)$ es borrado si y sólo si existe un elemento en $B^*(t)$ que lo domine. En consecuencia, un elemento de $\mathcal{M}(\mathcal{F}, \preceq)$ será un elemento de la sucesión $f(A(t))_{t\geq\tau}$ tan pronto como entre en el conjunto $f(A(\tau))$. Los elementos de $C(t)$ son incomparables con todos los miembros de $A(t)$ y de él se tomará necesariamente el número de elementos posible para no exceder el tamaño fijo m del conjunto $A(t)$ (mediante la función extrae).

Finalmente, si el conjunto A_t llegase a contener un elemento no óptimo, la irreducibidad de la cadena de Markov asegura que todos los elementos minimales serán generados en tiempo finito eliminando así aquellos elementos no óptimos que pudieran haber ingresado al conjunto A_t. ■

El Algoritmo 2 tiene dos detalles a discutir:

1. El proceso **Si** de la línea 11 debería aparecer como la parte **si no** del proceso **Si** de la línea 10:

   El conjunto D_b contiene a todos aquellos elementos de $A(t)$ a los cuales b domina. Si $D_b \neq \emptyset$ quiere decir que existen elementos en $A(t)$ a los cuales b domina, pero no sólo eso puesto que al haber elementos dominados por b en el conjunto $A(t)$, se concluye inmediatamente que b no puede ser dominado por algún elemento de $A(t)$. Esto se debe a que la relación de dominancia es transitiva y por construcción el conjunto $A(t)$ es una anticadena (todos sus elementos son incomparables): supongamos que existen $x, y \in A(t)$ tales que $x \prec b$ y $b \prec y$, tenemos entonces que $x \prec y$, una contradicción.

   De esta manera, el llevar a cabo siempre ambas líneas 10 y 11 es innecesario e implica desde un punto de vista práctico, demasiado trabajo extra.

   En [44] Rudolph presenta nuevamente este algoritmo con este detalle corregido.

2. Asumiendo que la matriz de transición que representa el algoritmo es irreducible, un elemento óptimo b^* puede ser generado varias veces en tiempo finito, pero es posible perderlo varias veces también pues en el caso de que $D_{b^*} = \emptyset$, es decir, que no domine a ningún elemento de $A(t)$, se colocará en el conjunto $C(t+1)$. En tal caso no se tiene la seguridad de que b^* entre en el conjunto $A(t+1)$ y de hecho si $|A(t+1)| = m$, b^* no entrará aún habiendo elementos no óptimos pero incomparables con b^*.

Dado lo anterior, a continuación propongo el Algorimo I que corrige el primer detalle antes mencionado. Sin embargo, al tratar de mejorar el segundo se sacrifica la convergencia al frente de Pareto:

1) $B(0)$ es generado aleatoriamente a partir de $\mathcal{S} = \mathbb{B}^{nl}$
2) $A(0) = \mathcal{M}_f(B(0), \preceq)$
3) $t = 0$
4) **repetir**
5) $B(t+1) = \text{genera}(B(t))$
6) $B^*(t+1) = \mathcal{M}_f(B(t+1), \preceq)$
7) $C(t+1) = \emptyset$
8) **para_cada** $b \in B^*(t)$
9) $D_b = \{a \in A(t) : f(b) \prec f(a)\}$
10) **si** $D_b \neq \emptyset$ **entonces** $A(t) \leftarrow (A(t) \setminus D_b) \cup \{b\}$
11) **si no: si** $\forall a \in A(t) : f(a)||f(b)$ **entonces** $C(t+1) \leftarrow C(t+1) \cup \{b\}$
12) **fin de para_cada**
13) $k = \min\{m, |A(t) \cup C(t+1)|\}$
14) $A(t+1) = \text{extrae}(k, A(t) \cup C(t+1))$
15) $t \leftarrow t+1$
16) **hasta** Criterio de terminación satisfecho

En este nuevo algoritmo la línea 11 es complementaria a la línea 10 corrigiendo el detalle descrito anteriormente. Sin embargo, por otra parte, puesto que se hace uso de la función *extrae* para obtener los mejores elementos de entre los que se tienen hasta el momento ($A(t)$) y los recién obtenidos ($C(t+1)$), la convergencia al frente de Pareto no puede ser asegurada:

- Por construcción, el conjunto $A(t)$ es una anticadena, es decir, tal conjunto siempre constará de individuos incomparables.

- Sin embargo, elementos óptimos pueden perderse a través de la función *extrae*, pues aunque todo el conjunto conste de elementos óptimos (lo cual se desconoce), algunos pueden perderse por posibles preferencias de la función, como mejor distribución o mayor alcance. Aunque en un momento dado el conjunto $A(t)$ conste de elementos minimales, la irreducibilidad de la matriz de transición correspondiente asegura la pérdida de alguno de ellos por los motivos antes mencionados. Este tipo de problemas han sido discutidos por Hanne en [25].

- A pesar de este detalle, este mecanismo se apega mucho más a los algoritmos implementados en la práctica. Este punto se tocará nuevamente en la siguiente sección.

Finalmente, en los casos de los Algoritmos 1 y 2, pudimos ver que la matriz de transición final que representa a la función genera() es el producto de las tres matrices de transición que describen los efectos estocásticos de la cruza, la mutación y la selección ($\mathbf{P} = \mathbf{CMS}$). De esta manera, para saber si un algoritmo en cuestión converge o no, lo único que hay que hacer es comprobar si la matriz correspondiente cumple con la propiedad de ser irreducible.

Cabe mencionar que algoritmos como MOGA [18], NPGA [28] y NSGA [49], de primera generación, pueden ser adaptados a este tipo de elitismo y converger al frente de Pareto.

6.3.2. Elitismo $\mu + \lambda$

Los dos algoritmos restantes presentados por Rudolph & Agapie en [45] se caracterizan por lo siguiente:

- **Algoritmo 3.** La población externa además de almacenar a los elementos minimales encontrados a lo largo de todo el proceso evolutivo se usa como el conjunto de padres para dar lugar a la población siguiente. El tamaño de la población externa puede variar hasta converger al número de elementos minimales que existan en el espacio de búsqueda.

- **Algoritmo 4.** Igual que el caso anterior pero restringiendo el tamaño de la población externa a un límite fijo.

Dadas las características anteriores, puede verse que ambos algoritmos se relacionan con el esquema de selección $\mu + \lambda$, pero no corresponden en su totalidad con él.

El Algoritmo 3 es el siguiente:

$B(0)$ es generado aleatoriamente a partir de $\mathcal{S} = \mathbb{B}^{nl}$
$A(0) = \mathcal{M}_f(B(0), \preceq)$
$t = 0$
repetir
$B(t+1) = \text{genera}(A(t))$
$A(t+1) = \mathcal{M}_f(A(t) \cup B(t+1), \preceq)$
$t \leftarrow t + 1$
hasta Criterio de terminación satisfecho

El Algoritmo 4 es el siguiente:

$B(0)$ es generado aleatoriamente a partir de $\mathcal{S} = \mathbb{B}^{nl}$
$A(0) = \mathcal{M}_f(B(0), \preceq)$
$t = 0$
repetir
  $B(t+1) = \text{genera}(A(t))$
  $B^*(t+1) = \mathcal{M}_f(B(t+1), \preceq)$
  $C(t+1) = \emptyset$
  para_cada $b \in B^*(t)$
    $D_b = \{a \in A(t) : f(b) \prec f(a)\}$
    si $D_b \neq \emptyset$ **entonces** $A(t) \leftarrow (A(t) \setminus D_b) \cup \{b\}$
    si $\forall a \in A(t) : f(a) || f(b)$ **entonces** $C(t+1) \leftarrow C(t+1) \cup \{b\}$
  fin de para_cada
  $k = \min\{m - |A(t)|, |C(t+1)|\}$
  $A(t+1) = A(t) \cup \text{extrae}(k, C(t+1))$
  $t \leftarrow t+1$
hasta Criterio de terminación satisfecho

Como podemos ver, los algoritmos 3 y 4 son muy similares a los algoritmos 1 y 2, pues la única diferencia es el argumento de la función *genera*, que en este caso es el conjunto $A(t)$ en lugar del conjunto $B(t)$. De esta manera, las demostraciones de la convergencia al frente de Pareto de los algoritmos 3 y 4 son análogas a las presentadas para los algoritmos 1 y 2, respectivamente. Sin embargo, en este caso la matriz de transición correspondiente a ambos algoritmos debe ser positiva, pues si fuese primitiva o irreducible, la convergencia no está asegurada: Supongamos que $l = 3$, es decir, $\mathbb{B}^{l=3} = \{0,1\}^3$, y que el conjunto $f(\mathbb{B}^3)$ cumple que:

$$\begin{tikzcd}
f(000) \arrow[r, "\prec"] & f(110) \arrow[r, "\prec"] & f(010) \arrow[r, "\prec"] & f(100) \\
f(001) \arrow[r, "\prec"] & f(111) \arrow[r, "\prec"] & f(101) \arrow[r, "\prec"] \arrow[ur, "\prec"] & f(011)
\end{tikzcd}$$

Así, $f(000)$ y $f(001)$ son elementos minimales. Ahora, supongamos que $A(t) = \{110, 111\}$ para algún $t \geq 0$ y que la matriz de transición que representa a la función *genera* es irreducible o primitiva. Esto implica que sólo puede invertir un bit a la vez en la población, invertir un solo bit en un individuo o bien dejarlo intacto. En este caso, el conjunto $B(t+1)$ sólo puede contener elementos del conjunto $\{010, 011, 100, 101, 110, 111\}$ por lo que $A(t+1) = A(t) = \{110, 111\}$ en la siguiente iteración. En este caso el conjunto de elementos minimales es inalcanzable.

De lo anterior, podemos ver claramente que en los casos de los algoritmos 3 y 4 es necesaria la condición de que la matriz de transición sea positiva, es decir, que en cada iteración, la probabilidad de generar cualquier población (en particular un individuo determinado) sea estrictamente positiva.

Ahora bien, el Algoritmo 3 se relaciona con el esquema $\mu + \lambda$ pues a partir de μ individuos se generan λ y posteriormente, a partir de los $\mu + \lambda$ se elige a los mejores elementos (los no dominados), los cuales fungirán como padres en la siguiente iteración. Sin embargo, claramente el conjunto de padres $A(t)$ no tiene el tamaño fijo μ, sino que la cardinalidad $|A(t)|$ del conjunto $A(t)$ variará hasta tomar el valor (en el límite) del número de elementos minimales que existan en el espacio de búsqueda.

El Algoritmo 4 está aún más relacionado con el esquema $\mu + \lambda$ pues fija el tamaño máximo de la población de padres $A(t)$ a una constante m. Si consideramos que $m = \mu$, el Algoritmo 4 se asegura de que la población de padres no tenga más de μ elementos. Sin embargo, este algoritmo padece de los mismos detalles que se discutieron para el Algoritmo 2.

Dado que ninguno de los dos algoritmos anteriores se apega completamente al esquema de selección $\mu + \lambda$, en [43] Rudolph presenta un algoritmo que cumple tal condición, lo llamaremos Algoritmo 5:

$B(0)$ es generado aleatoriamente a partir de $\mathcal{S} = \mathbb{B}^{nl}$
$A(0) = \mathcal{M}_f(B(0), \preceq)$
$t = 0$
repetir
  $B(t+1) =$ genera$(A(t))$
  $B^*(t+1) = \mathcal{M}_f(B(t+1), \preceq)$
  $B(t+1) \leftarrow B(t+1) \setminus B^*(t+1)$
  $C(t+1) = \emptyset$
  para_cada $b \in B^*(t)$
    $D_b = \{a \in A(t) : f(b) \prec f(a)\}$
    si $D_b \neq \emptyset$
    entonces $A(t) \leftarrow (A(t) \setminus D_b) \cup \{b\}$, $B^*(t+1) \leftarrow B^*(t+1) \setminus \{b\}$
    si no: si $\forall a \in A(t) : f(a) || f(b)$
      entonces $C(t+1) \leftarrow C(t+1) \cup \{b\}$,$B^*(t+1) \leftarrow B^*(t+1) \setminus \{b\}$
  fin de para_cada
  $k = \min\{m - |A(t)|, |C(t+1)|\}$
  $A(t+1) = A(t) \cup \text{extrae}(k, C(t+1))$
  $k = \min\{m - |A(t)|, |B^*(t+1)|\}$
  $A(t+1) = A(t) \cup \text{extrae}(k, B^*(t+1))$
  $k = \min\{m - |A(t)|, |B(t+1)|\}$
  $A(t+1) = A(t) \cup \text{extrae}(k, B(t+1))$
  $t \leftarrow t+1$
hasta Criterio de terminación satisfecho

Este algoritmo pierde la invariante de que el conjunto $A(t)$ es una anticadena en cada iteración, pues el mantener fijo su tamaño trae consigo la posibilidad de que entren elementos dominados a él. Sin embargo, la convergencia puede demostrarse de manera similar a los algoritmos propuestos por Rudolph antes descritos.

Por otra parte, el Algoritmo 5 sigue teniendo el detalle 2 que discutimos anteriormente.

A continuación propongo el Algoritmo II, que se apega al esquema de selección $\mu + \lambda$ y que corrige el detalle que en el Algoritmo 5 sigue fallando:

$B(0)$ tal que $|B(0)| = \mu$, es generado aleatoriamente a partir de $\mathcal{S} = \mathbb{B}^{nl}$

$A(0) = \mathcal{M}_f(B(0), \preceq)$

$t = 0$

repetir

$B(t+1) = \text{genera}(B(t))$, $|B(t+1)| = \lambda$

$B^*(t+1) = \mathcal{M}_f(B(t+1), \preceq)$

$C(t+1) = \emptyset$

para_cada $b \in B^*(t)$

$D_b = \{a \in A(t) : f(b) \prec f(a)\}$

si $D_b \neq \emptyset$ **entonces** $A(t) \leftarrow (A(t) \setminus D_b) \cup \{b\}$

si no: si $\forall a \in A(t) : f(a) || f(b)$ **entonces** $C(t+1) \leftarrow C(t+1) \cup \{b\}$

fin de para_cada

$k = \min\{\mu, |A(t) \cup C(t+1)|\}$

$A(t+1) = \text{extrae}(k, A(t) \cup C(t+1))$

si $|A(t+1)| = \mu$ **entonces** $B(t+1) = A(t+1)$

si no $B(t+1) = A(t+1) \cup \text{extrae2}(\mu - |A(t+1)|, B(t) \setminus B^*(t+1))$

$t \leftarrow t+1$

hasta Criterio de terminación satisfecho

Por construcción, el conjunto A_t es una anticadena. La diferencia de este algoritmo con el Algoritmo 3 es que, dado que el tamaño de la población de padres debe ser siempre igual a μ, se tienen dos casos posibles:

1. Si $|A(t+1)| \geq \mu$, nos encontramos en el caso discutido anteriormente para el Algoritmo 2 en el que algunos elementos óptimos pueden perderse a través de la función *extrae*.

2. Si $|A(t+1)| < \mu$, la población de padres $B(t+1)$ debe completarse escogiendo a los mejores elementos no minimales de la población recién generada, a través de algún mecanismo arbitrario representado por la función $extrae2$.

De esta manera, en el mejor de los casos, es decir, el caso 1, la convergencia nuevamente no puede asegurarse por las mismas razones que en el Algoritmo 2. Sin embargo, como en tal caso, el Algoritmo II representa la manera más común de implementar el elitismo $\mu + \lambda$.

Conclusiones y Trabajo Futuro

En el presente trabajo de tesis se llevó a cabo un estudio teórico principalmente de la convergencia de los algoritmos evolutivos y, de manera muy particular, de los Algoritmos Genéticos.

Se mostró la convergencia del AG Simple para posteriormente estudiar la complejidad de estimar el tiempo esperado de convergencia de dicho algoritmo. Respecto a este punto, pudimos ver que aunque el modelo teórico es correcto, resulta muy complicado obtener una expresión en términos de los parámetros del AG, que acote el número de iteraciones necesarias para alcanzar el óptimo. La dificultad se debe a las dimensiones de la matriz de transición correspondiente al proceso evolutivo la cual, conforme aumenta el valor de los parámetros, crece exponencialmente. Esto significa que para poder predecir el comportamiento del algoritmo, se hace necesario conocer la matriz correspondiente, lo cual en un caso real resulta en un proceso muy costoso. En este caso, definitivamente la alternativa es usar otro tipo de herramientas teóricas para modelar el AG que no requieran tanto conocimiento a priori para poder predecir algunos aspectos de su comportamiento.

Por otra parte, se dio una breve introducción a los Algoritmos Evolutivos Multiobjetivo (AEMO), clasificándolos de acuerdo a si usan algún mecanismo de elitismo o no, siendo de primera generación los que no lo usan y de segunda los que sí. Posteriormente, se estudiaron algunos modelos teóricos de algoritmos de ambas generaciones para concluir cuáles y bajo qué condiciones convergen al frente de Pareto.

Como pudimos ver, los AEMOS's de primera generación, es decir, aquellos que no usan ningún mecanismo de elitismo, no convergen al frente de Pareto. Al menos no lo hacen si la convergencia está condicionada a que todos los miembros de la población sean óptimos de Pareto. De esta manera, lo que resulta interesante estudiar de estos algoritmos es, si bien no convergen al frente de Pareto, ¿cuántos elementos de la población logran ser óptimos?, es decir, ¿qué porcentaje de la población llega al frente de Pareto?, ¿cuál es el tiempo esperado para obtener la primera solución óptima? En este sentido, se propone que lo que corresponde es estudiar la dinámica poblacional, es decir, de qué manera los mecanismos de compartición de aptitud usados por los algoritmos de esta generación, afectan la distribución de los individuos a lo largo del proceso evolutivo ya que su fin es la dispersión de éstos a lo largo del frente de Pareto.

Respecto a los AEMOS's de segunda generación, pudimos ver que en general el uso de una población externa en la que se almacenen los elementos minimales encontrados durante el proceso, asegura la convergencia de esta última al frente de Pareto, bajo ciertos procesos de actualización. En este sentido, es muy interesante analizar hasta qué punto tales procesos de actualización se asemejan a los implementados en la práctica y determinar entonces en qué casos la convergencia se preserva. Tal es el caso del Algoritmo I propuesto en el Capítulo 6 que intenta modelar el proceso más común de actualización de la población externa y cuya convergencia al frente de Pareto, sin embargo, no puede asegurarse.

En el caso de los AEMO's que usan el esquema de selección $\mu + \lambda$ como mecanismo elitista, se mostraron cinco algoritmos propuestos por Rudolph [45, 43] de los cuales sólo uno se apega a tal esquema y cuya convergencia está asegurada. Sin embargo, en este caso nuevamente nos encontramos ante la situación de que tales modelos no representan aquellos algoritmos que en la práctica son usados para implementar tal forma de elitismo. De tal manera, se propuso el Algoritmo II cuyo mecanismo de elitismo es muy similar a los que comúnmente implementan el esquema de selección $\mu + \lambda$. Sin embargo, como en el caso del Algoritmo I, la convergencia no puede ser asegurada.

Así pues, en el campo de los Algoritmos Evolutivos en general, aún quedan varias preguntas de carácter teórico por responder, además de que como se mencionó en la introducción de este trabajo, los modelos estudiados están basados en Cadenas de Markov principalmente, de manera que es posible utilizar herramientas distintas mediante las cuales el modelado pueda resultar más simple o posiblemente pueda dar lugar a conclusiones más útiles en la práctica.

Bibliografía

[1] Alexandru Agapie. Genetic algorithms: Minimal conditions for convergence. In *Artificial Evolution European Conference, AE 97*, pages 183–206, Nîmes, France, 1997. Springer Verlag.

[2] Carol A. Ankenbrandt. An extension to the theory of convergence and a proof of the time complexity of genetic algorithms. In Gregory J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 53–68. Morgan Kaufmann Publishers, 1991.

[3] Thomas Bäck, editor. *Evolutionary Algorithms in Theory and Practice.* Oxford University Press, New York, 1996.

[4] J. E. Baker. Reducing bias and inefficiency in the selection algorithm. In John Grefenstette, editor, *Proceedings of the Second International Conference on Genetic Algorithms*, pages 14–21, Hillsdale, New Jersey, 1987. Lawrence Erlbaum Associates.

[5] L. B. Booker. Intelligent behavior as an adaptation to the task environment. Technical Report 243, University of Michigan at Ann Arbor, Ann Arbor, Michigan, 1982.

[6] A. Brindle. *Genetic Algorithms for Function Optimization.* PhD thesis, Department of Computer Science of the University of Alberta, Alberta, Canada, 1981.

[7] Y. S. Chow and H. Teicher. *Probability Theory.* Springer, New York, 1978.

[8] Carlos A. Coello Coello and Gregorio Toscano Pulido. A micro-genetic algorithm for multiobjective optimization. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 126–140. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[9] David W. Corne, Nick R. Jerram, Joshua D. Knowles, and Martin J. Oates. PESA-II: Region-based selection in evolutionary multiobjective optimization. In Lee Spector, Erik D. Goodman, Annie Wu, W.B. Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*, pages 283–290, San Francisco, California, 2001. Morgan Kaufmann Publishers.

[10] David W. Corne, Joshua D. Knowles, and Martin J. Oates. The pareto envelope-based selection algorithm for multiobjective optimization. In Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 839–848, Paris, France, 2000. Springer. Lecture Notes in Computer Science No. 1917.

[11] Charles Darwin. *The Origin of Species by Means of Natural Selection or the Preservation of Favored Races in the Struggle for Life.* The Book League of America, 1929. Originally published in 1859.

[12] Kalyanmoy Deb, Samir Agrawal, Amrit Pratab, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 849–858, Paris, France, 2000. Springer. Lecture Notes in Computer Science No. 1917.

[13] Kalyanmoy Deb and David E. Goldberg. An investigation of niche and species formation in genetic function optimization. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 42–50, San Mateo, California, June 1989. George Mason University, Morgan Kaufmann Publishers.

[14] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[15] Mark Erickson, Alex Mayer, and Jeffrey Horn. The Niched Pareto Genetic Algorithm 2 applied to the design of groundwater remediation systems. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 681–695. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[16] David B. Fogel. *Evolutionary Computation. Toward a New Philosophy of Machine Intelligence.* The Institute of Electrical and Electronic Engineers, New York, 1995.

[17] Lawrence J. Fogel. *Artificial Intelligence through Simulated Evolution.* John Wiley, New York, 1966.

[18] Carlos M. Fonseca and Peter J. Fleming. Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.

[19] M. P. Fourman. Compaction of symbolic layout using genetic algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 141–153. Lawrence Erlbaum, 1985.

[20] A. S. Fraser. Simulation of genetic systems by automatic digital computers. *Australian Journal of Biological Sciences*, (10):484–499, 1957.

[21] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49. Lawrence Erlbaum Associates, 1987.

[22] D. E. Goldberg and P. Segrest. Finite markov chain analysis of genetic algorithms. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 1–8. Lawrence Erlbaum Associates, 1987.

[23] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Publishing Co., Reading, Massachusetts, 1989.

[24] P. Hajela and C. Y. Lin. Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4:99–107, 1992.

[25] Thomas Hanne. On the convergence of multiobjective evolutionary algorithms. *European Journal of Operational Research*, 117(3):553–564, 1999.

[26] John H. Holland. *Adaptation in Natural and Artificial Systems.* Ann Arbor : University of Michigan Press, 1975.

[27] Jeffrey Horn. Finite markov chain analysis of genetic algorithms with niching. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 110–117. Morgan Kaufmann Publishers, 1993.

[28] Jeffrey Horn, Nicholas Nafpliotis, and David E. Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, volume 1, pages 82–87, Piscataway, New Jersey, June 1994. IEEE Service Center.

[29] M. Iosifescu. *Finite Markov Processes and Their Applications.* Wiley, Chichester, 1980.

[30] W. Jakob, M. Gorges-Schleuter, and C. Blume. Application of genetic algorithms to task planning and learning. In R. Männer and B. Manderick, editors, *Parallel Problem Solving from Nature, 2nd Workshop*, Lecture Notes in Computer Science, pages 291–300, Amsterdam, 1992. North-Holland Publishing Company.

[31] A. K. De Jong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems.* PhD thesis, University of Michigan, 1975.

[32] John G. Kemeny and J. Laurie Snell. *Finite Markov Chains.* D. Van Nostrand Company, Inc., Princeton, New Jersey, 1960.

[33] Joshua D. Knowles and David W. Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

[34] Jean Baptiste Lamarck, editor. *Zoological Philosophy: An Exposition with Regard to the Natural History of Animals.* Chicago University Press, Chicago, 1984 (Publicado originalmente en 1809 en francés, como *Philosophie Zoologique*).

[35] Sushil J. Louis and Gregory J.E. Rawlins. Syntactic analysis of convergence in genetic algorithms. In L. Darrell Whitley, editor, *Foundations of Genetic Algorithms*, pages 141–151, San Mateo, CA, 1993. Morgan Kaufmann Publishers.

[36] S.W. Mahfoud. *Niching Methods for Genetic Algorithms.* PhD thesis, University of Illinois, Urbana-Champaign, 1995.

[37] Gregor Johann Mendel. Experiments in plant hybridisation. *Journal of the Royal Horticultural Society*, (26):1–32, 1901.

[38] J.Ñ. Morse. Reducing the size of the nondominated set: Pruning by clustering. *Computers and Operations Research*, 7(1-2):55–66, 1980.

[39] Allen E. Nix and Michael D. Vose. Modeling genetic algorithms with markov chains. *Annals of Mathematics and Artificial Intelligence*, 5:79–88, 1992.

[40] A. Prügel-Bennett and J.L. Shapiro. An analysis of genetic algorithms using statistical mechanics. *Physical Review Letters*, 72(9):1305–1309, 1994.

[41] Günter Rudolph. Convergence properties of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, 1(5):96–101, 1994.

[42] Günter Rudolph. *Convergence Properties of Evolutionary Algorithms*. Verlag Dr. Kovač, Hamburg, 1997.

[43] Günter Rudolph. Evolutionary search under partially ordered fitness sets. In *Proceedings of the International NAISO Congress on Information Science Innovations (ISI 2001)*, pages 818–822. ICSC Academic Press: Millet/Sliedrecht, 2001.

[44] Günter Rudolph. Some theoretical properties of evolutionary algorithms under partially ordered fitness values. In C. Fabian and I. Intorsureanu, editors, *Proceedings of the Evolutionary Algorithms Workshop (EAW-2001)*, pages 9–22, Bucharest, Romania, 2001.

[45] Günter Rudolph and Alexandru Agapie. Convergence properties of some multi-objective evolutionary algorithms. In *Proceedings of the 2000 Conference on Evolutionary Computation*, pages 1010–1016, Piscataway, New Jersey, 2000. IEEE.

[46] J. David Schaffer. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. PhD thesis, Vanderbilt University, 1984.

[47] J. David Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum, 1985.

[48] Hans-Paul Schwefel. *Numerical Optimization of Computer Models*. John Wiley & Sons, Great Britain, 1981.

[49] N. Srinivas and Kalyanmoy Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, Fall 1994.

[50] Jürgen Teich, Eckart Zitzler, and Shuvra S. Bhattacharyya. 3D Exploration of software schedules for DSP algorithms. In *7th International Workshop on Hardware/Software Codesign (CODES'99)*, pages 168–172, May 1999.

[51] Michael D. Vose. Modeling simple genetic algorithms. In L. D. Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 63–73, San Mateo California, 1991. Morgan Kaufmann Publishers.

[52] Michael D. Vose. *The Simple Genetic Algorithm: foundations and theory.* MIT Press, Cambridge, Massachusetts, 1999.

[53] Hugo De Vries. *Species and Varieties: Their Origin by Mutation.* Open Court, Chicago, 1906.

[54] August Weismann, editor. *The Germ Plasm: A Theory of Heredity.* Scott, London, UK, 1893.

[55] A. Wetzel. Evaluation of the effectiveness of genetic algorithms in combinational optimization. Technical Report (no publicado), University of Pittsburgh, Pittsburgh, 1983.

[56] D. Whitley. The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In J. David Schaffer, editor, *Proceedings of the Third Conference on Genetic Algorithms*, pages 116–121, San Mateo, California, jun 1989. George Mason University, Morgan Kaufmann Publishers.

[57] P. B. Wilson and M. D. Macleod. Low implementation cost IIR digital filter design using genetic algorithms. In *IEE/IEEE Workshop on Natural Algorithms in Signal Processing*, pages 4/1–4/8, Chelmsford, U.K., 1993.

[58] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the strength pareto evolutionary algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, May 2001.