Centro de Investigación en
Matemáticas, A.C.

Optimización de Confiabilidad en Sistemas Multi-Estado

L.A. Lina Constanza Vargas Serdio

Julio 2010

Asesor: Dr. Enrique Villa Diharce

Índice general

Capítulo 1

Introducción

1.1. Antecedentes

Los sistemas que desempeñan alguna tarea en la vida real hoy en día se han vuelto cada vez más complejos. Dichos sistemas a los que se hace alución van desde aquellos que realizan un proceso en la industria dando como resultado algún producto en específico hasta los que forman parte de un sistema mayor, por ejemplo: circuitos electrónicos que pertenecen a algún aparato eléctrico, los componentes de una red computacional o un cluster, etc. Estos sistemas constan de una serie de elementos que se encuentran conectados de algún modo generando de esta forma toda una estructura compleja.

Dado este contexto, puede pensarse que cada elemento de cualquier sistema es una entidad compleja que puede a su vez estar formado por subsistemas con diversos elementos. Es así, como cada uno de los elementos forma parte importante del sistema entero, cooperando o contribuyendo al desempeño del mismo ya sea en proporciones iguales o distintas. Esto último hace referencia al desempeño por elemento como una parte importante de la cual depende el funcionamiento total esperado de un cierto sistema.

En torno al desempeño de cada elemento, puede pensarse que dichos componentes poseen diferentes modos de operación o de funcionamiento llamados .estados de desempeño", los cuales pueden ir desde el perfecto estado, es decir, aquel en el cual el elemento se encuentra en perfecto funcionamiento, hasta el estado de falla, en el cual el elemento no es apto para desempeñar más su labor y de este modo ya no contribuye más al desempeño del sistema. En este punto, un sistema que no posee todos sus elementos funcionando correctamente no es apto de proveer o satisfacer las demandas del cliente, generando insatisfacción y quizá demora de producción en otros casos. Esto se ve reflejado en el ambiente Industrial en la competitividad de las empresas para poder satisfacer siempre cierta demanda bajo un cierto costo.

Este tipo de sistemas poseen diferentes estados o niveles de desempeño en su totalidad. El número de estados que posean depende del número de estados que presenten los elementos que los conforman. Es decir, el sistema puede funcionar desde el perfecto estado hasta el estado de falla pasando por diferentes niveles de desempeño. Estos son llamados sistemas multi-estado.

Para poder mantener cierta competitividad en cuanto al funcionamiento y desempeño de los sistemas, las industrias o empresas se ven obligadas a introducir conceptos de confiabilidad en la producción o en el desarrollo de algunas actividades. En estos términos, la confiabilidad es comunmente conocida como una herramienta que ayuda a prevenir fallas, es decir, calidad a través del tiempo. La confiabilidad juega un papel muy importante para las industrias y los usuarios de cualquier sistema. Una confiabilidad alta incrementa los costos de producción pero asegura el buen rendimiento del sistema a través del tiempo, pudiendo cumplir con alta probabilidad una cierta demanda y mantenerse competitivos en los mercados correspondientes. Sin embargo, el mantener un cierto nivel de confiabilidad durante el tiempo de vida de un sistema resulta costoso y quizá complicado. En este punto se puede visualizar un problema de optimización, máximizar la confiabilidad del sistema minimizando el costo de dicha acción.

En este trabajo, se aborda el problema de proveer de una secuencia óptima de mantenimientos a un sistema específico tratando de mantener un nivel mínimo de confiabilidad y de cumplimiento dada cierta demanda. Se plantea el problema de optimización: minimizar el costo de una política de mantenimientos dadas dos restricciones, una con respecto a la confiabilidad y otra acerca del nivel de demanda a ser satisfecho. Se trabaja con un problema de mantenimiento por las siguientes razones:

- Cuando a cierto sistema se le deben realizar mantenimientos que consten de reemplazo de piezas o reparación de las mismas, muchas veces se debe detener por completo el proceso para poder llevar a cabo dichas acciones. Esto impacta directamente en la competitividad del producto o resultado y en el costo de perder tiempo de producción y el mismo del mantenimiento.

- Si se produce un plan de mantenimientos preventivos puede reducirse el costo de paro de producción y el asociado a esperar hasta que una pieza o elemento falle para repararla o reemplazarla.

- Además, un plan de mantenimientos preventivos promueve la creación de un plan estratégico de producción que incluya los mantenimientos.

Recientemente algunos autores como Lisnianski y Levitin han tratado los problemas de optimización de la confiabilidad de sistemas multi-estado aplicando algortimos

genéticos y aplicando la Función Generadora Universal (Universal Generating Function, UGF).

Así, en este trabajo se tratará la función generadora universal basada en el desempeño del sistema ya que ha sido demostrado que trabaja rápidamente en problemas cuyos algoritmos computacionales resultan largos y tardados.

1.2. Motivación

Este trabajo es motivado por una serie de ejemplos que proponen Lisnianski y Levitin [1]. Ellos tratan problemas de optimización de confiabilidad en sistemas multiestado con diferentes estructuras. Utilizan la función generadora universal y algoritmos genéticos para tal propósito. Sin embargo, algunos de los problemas que se abordan parecen ser más complejos de lo que puede pensarse que son, es decir, los métodos con los que han sido resueltos y las herramientas utilizadas para tal fin deben ser descritas y abordadas con mayor claridad. Por otra parte, los algoritmos computacionales aplicados para resolver el problema de optimización, pueden ser mejorados y así obtener mejores soluciones que las que se presentan.

Por otra parte, el poder establecer una política óptima de mantenimientos suele ser de gran utilidad en la planeación de la producción como se mencionó anteriormente. Particularmente en este trabajo, se hará un mayor énfasis en los mantenimientos imperfectos y en la determinación de los tiempos en los cuales se deben hacer estas actividades. Para las industrias que presentan sistemas de producción en los cuales es necesario planificar los mantenimientos, suele ser de gran utilidad el poder determinar secuencias y tiempos óptimos que impliquen un menor costo y aseguren mantener los niveles de confiabilidad para obtener máximos beneficios en cuanto a la producción.

Este tipo de problemas utilizan conceptos estadísticos y probabilísticos importantes, tales como cadenas de Markov, probabilidades límite, convergencias, funciones de renovación, valores esperados, distribución del tiempo de vida entre otros; los cuales permiten hacer del problema una aplicación importante de dichos conceptos mostrando en si mismo un grado de complejidad en cuanto a la abstracción que es necesaria para poder comprender y abordar dicho problema.

La aplicación computacional es también una parte muy interesante en este trabajo, pues al tratar de optimizar cierta función compleja que posee varias restricciones, las herramientas como algoritmos genéticos o las metaheurísticas resultan ser opciones viables para resolver dicho problema. Sus métodos de búsqueda intensivos, aunado al caso particular de Colonia de Hormigas (Ant Colony, ACO) aseguran la convergencia a la solución óptima tras un número suficientemente grande iteraciones por ser de naturaleza estocástica.

Es así, como al conjuntarse dichos conceptos, se abordó problemas de optimización de mantenimientos perfectos e imperfectos apoyados por el uso del algoritmo genético ACO y la programación en paralelo para mejorar la rapidez de los procesos generados al tratar de resolver el problema bajo las restricciones que serán propuestas posteriormente.

1.3. Objetivos

A continuación se presentan los objetivos más importantes dentro del trabajo:

- Confiabilidad en sistemas multi-estado: Introducir al estudio de la confiabilidad en sistemas multi-estado y la importancia de mantener un nivel dado durante el tiempo de vida del mismo. Mostrar la relación que existe entre una cierta demanda y el desempeño del sistema, así como la importancia que tiene el poder planificar una secuencia o política óptima de mantenimientos para establecer un buen funcionamiento del mismo con el mejor rendimiento y el costo mínimo. Se tratarán dos tipos de problemas, aquellos sistemas que necesitan de mantenimientos perfectos en un tiempo específico para todos sus elementos y aquellos que necesitan de mantenimientos preventivos en diferentes tiempos aplicados a distintos elementos durante el ciclo de vida del sistema. Para ambos casos, se tomará un ejemplo de Lisnianski y Levitin [1] y se propondrán políticas de mantenimiento tratando de replicar o mejorar las soluciones de los autores.

- Tiempos de mantenimientos: En el caso de mantenimientos imperfectos, se hará un énfasis importante en la obtención correcta de los tiempos idóneos en los cuales es viable y necesario implementar cierta actividad preventiva para evitar que los niveles de confiabilidad del sistema sean menores a los requeridos y para proveer siempre al mercado del nivel de demanda solicitado.

- Colonia de Hormigas (ACO): Se desea aplicar ACO a este tipo de problemas, sabiendo que hay muy pocas aplicaciones en este campo, en especial en la creación de secuencias de mantenimientos. Uno de los objetivos más importantes en esta parte es el lograr implementar este algoritmo de tal modo que toda solución que se obtenga por agente, sea factible.

- Programación en paralelo: El objetivo es implementar dicho modo de programación para mejorar el desempeño de las máquinas utilizadas y obtener las soluciones en un mejor tiempo, aprovechando cada uno de los procesadores disponibles.

1.4. Contribuciones

Este trabajo contribuye en los campos de la confiabilidad de sistemas multi-estado y en la programación en paralelo en R [7] mostrando la facilidad para realizar tal aplicación. Algunas de las contribuciones más importantes son:

- La introducción de un algoritmo genético de naturaleza estocástica para la resolución del problema de mantenimientos imperfectos. La propuesta de soluciones óptimas y mejoradas tales que cumplan todas las restricciones dadas y generen una idea del cómo puden ser aplicado dicho algoritmo a diferentes problemas que posean una naturaleza similar.
- Mostrar una aplicación más del algoritmo Ant Colony Optimization. De este modo verificar la versatilidad del mismo.
- Mostrar la facilidad de la programación en paralelo en R [7] y el cómo mejora los procesos que pueden ser realizados en dicho lenguaje de programación.

1.5. Contenido

- El capítulo 2 proporciona a un nivel general, los conceptos básicos sobre la confiabilidad de sistemas multi-estado y el desempeño de los mismos. Describe las funciones de estructura de dichos sistemas así como el uso de la Función Generadora Universal y su utilidad en el cálculo de índices que midan el comportamiento de los elementos y del conjunto en su totalidad. Este mismo capítulo trata sobre las generalidades y bases teóricas que requiere ACO.
- El capítulo 3 contiene el desarrollo de la metodología a seguir para la resolución de cada uno de los dos diferentes tipos de problemas, haciendo énfasis en las consideraciones que se hicieron y las propuestas para aplicar las diversas herramientas para obtener solución a los problemas propuestos.
- El capítulo 4 presenta los resultados obtenidos y comparaciones entre los propios de los autores así como herramientas gráficas que ayudan en la comprensión de la naturaleza del problema.
- El capítulo 5 contiene las conclusiones generales y recomendaciones así como trabajo a futuro.
- Un apédice que contiene una breve explicación de la programación en paralelo en R [7].

Capítulo 2

Preliminares

2.1. Conceptos Básicos de Sistemas Multi-Estado

2.1.1. Definiciones principales y propiedades

Todo sistema tecnológico se encuentra diseñado para desemepeñar ciertas actividades en un ambiente dado. Algunos sistemas pueden llevar a cabo sus actividades con varios niveles de desempeño o de eficiencia usualmente llamados *tasas o niveles de desempeño.* Un sistema que posee un número finito de niveles de desempeño es llamado un sistema multi-estado (MSS multi-state system). Usualmente dichos sistemas se encuentran compuestos que pueden ser por si mismos multi-estado.

El caso más simple de MSS es un sistema Binario así como el caso más simple de elementos multi-estado. Existen diversas situaciones en las cuales un sistema puede ser considerado MSS por ejemplo:

1. Cualquier sistema que consta de diferentes unidades con efecto cumulativo sobre el desempeño total del sistema. El nivel de desempeño de un sistema dependerá de la disponibilidad de sus elementos.

2. El nivel de desempeño de los elementos que componen un sistema puede variar como resultado de su deterioro o por condiciones variables del ambiente en el que se encuentran. Las fallas de los elementos de MSS aportan a la degradación del desempeño del sistema completo.

Estos son solo dos ejemplos para motivar el pensamiento de qué sistemas pueden ser considerados MSS o qué elementos pueden ser binarios o multi-estado. Para el estudio de MSS se definirán algunos conceptos básicos en el desarrollo de este estudio.

2.1.2. Elementos binarios y multi-estado

Consideremos un elemento como una unidad que pertenece a un sistema que no puede ser sub-dividida. Un elemento binario posee dos estados: perfecto funcionamiento y falla completa. Cualquier artículo es considerado en perfecto funcionamiento al tiempo $t = 0$. El estado en el cuál se encuentra el artículo al tiempo t se encuentra definido por la variable aleatoria:

$$X(t) = \begin{cases} 1, & funcionando \\ 0, & en \quad falla \end{cases}.$$

Como puede observarse, la variable aleatória $X(t)$ depende del tiempo t en el que se encuentre el elemento, así un concepto importante a considerar es el tiempo al cuál el artículo cambia de estado; este es llamado "tiempo de falla". El tiempo de falla no es necesariamente medido en unidades de tiempo, puede ser medido mediante ciclos, distancias, conteos, etc.

Sea T la variable aleatória que define el tiempo de falla de cualquier elemento MSS. Asumiendo que T es una variable aletoria contínua, la distribución de T está dada por:

$$F(t) = P(T \leq t) = \int_0^t f(u)\, du,$$

donde $f(u)$ es la función de densidad de T. Así, dada la función de distribución podemos obtener la probabilidad de que el artículo falle en un intervalo de tiempo $(0, t]$. Se pueden obtener expresiones para el valor esperado de la vida del elemento, $E(T)$, y para la función de Confiabilidad $C(t) = 1 - F(t)$.

Si mantenemos el supuesto de que los elementos dentro de un sistema solamente tienen dos estados, podría generar problemas en el análisis de unidades con un comportamiento decreciente, en el sentido que puede haber artículos que sigan funcionando aún sin llegar al estado de falla pero que han sufrido cambios en su estructura de tal modo que no se encuentran exactamente como en $t = 0$.

Un elemento multi-estado posee diferentes niveles de desempeño o .estados.a lo largo de su vida útil o del periodo de funcionamiento que posea. Se asume que en $t = 0$, el artículo se encuentra en el mejor estado, o en el estado de funcionamiento perfecto. Las fallas que sufre hacen que el desempeño o la calidad del funcionamiento de dicho artículo decrezcan.

Es así como el estado de los artículos multi-estado al tiempo t puede ser descrito por una variable aleatoria. discreta $G(t)$ que toma valores dentro de un conjunto:

$$g(t) = \{g_0(t), ..., g_n(t)\}.$$

Generalmente $g_0(t)$ representa el estado de falla completa del elemento. En cualquier momento que se tengan cambios en el desempeño, se dice que se tiene üna transición de

estado.[en] el componente. Por otra parte, las probabilidades asociadas a los diferentes estados del elemento son representadas por el conjunto:

$$p(t) = \{p_0(t), ..., p_n(t)\},$$

donde $p_i(t)$ es la probabilidad de que la variable $G(t)$ se encuentre en el estado $g_i(t)$ al tiempo t.

$$p_i(t) = P[G(t) = g_i(t)].$$

Si se tienen n diferentes elementos dentro de un sistema, cada elemento j poseerá un nivel de desempeño $G_j(t)$ en cualquier instante $t \geq 0$, el cual tomará valores dentro de un conjunto

$$g_j = \left\{g_{j1}, g_{j2}..., g_{jk_j}\right\},$$

donde k_j es el número de estados diferentes que posee el elemento j, g_{ji} es la tasa de desempeño correspondiente a dicho elemento en el estado i, $i \in \{1, 2, ..., k_j\}$.

Así mismo, las probabilidades asociadas con los diferentes estados o niveles de desempeño de cada elemento de un sistema al tiempo t están representadas por el conjunto

$$p_j(t) = \left\{p_{j1}(t), p_{j2}(t) ..., p_{jk_j}(t)\right\},$$

donde

$$p_{ji}(t) = \{PG_j(t) = g_{ji}\}.$$

Nótese que un elemento puede estar sólamente en un estado al tiempo t. Esto significa que los estados por componente conforman un conjunto de eventos mutuamente exluyentes tales que $\sum_{i=0}^{k_j} p_{ji}(t) = 1$ para todo $0 \leq t \leq T$. La colección de pares ordenados $(g_{ji}, p_{ji}(t))$, $i = 0, 1, 2, ..., k_j$. determinan completamente la distribución de probabilidad asociada al desempeño por elemento multi-estado.

2.1.3. Definiciones de MSS

Dado un MSS con n elementos, las tasas o niveles de desempeño son determinadas por las correspondientes a cada uno de sus elementos. En cada momento, los elementos del sistema poseen ciertos niveles de desempeño correspondientes a cada uno de sus estados. El estado del sistema entero tiene K diferentes estados con g_i tasas de desempeño correspondientes a cada estado, $i \in \{1, ..., K\}$. Las tasas de desemepeño de MSS al tiempo t toma valores del conjunto $\{g_1, ..., g_K\}$.

Sea $L^n = \{g_{11}, g_{12}..., g_{1k_1}\} \times \{g_{21}, g_{22}..., g_{2k_2}\} \times ... \times \{g_{n1}, g_{n2}..., g_{nk_n}\}$ el espacio de todas las posibles combinaciones de tasas de desempeño de todos los elementos del sistema y $M = \{g_1, ..., g_K\}$ es el espacio de todos los valores posibles de tasas de

desempeño para el sistema completo. Se define la transformación $\phi\left(G_1\left(t\right),...,G_n\left(t\right)\right): L^n \rightarrow M$, que mapea el espacio de todas las posibles combinaciones de las tasas de desempeño de los elementos del sistema en el espacio de tasas de desempeño del sistema entero. A esta transformación se le llama *Función de Estructura*. La única diferencia se encuentra en la definición de los estados del estado, por ejemplo, una función estructura binária es mapeada $\{0,1\}^n \rightarrow \{0,1\}$, mientras que MSS posee espacios mucho más complejos.

Se puede definir un modelo genérico correspondiente a un MSS. Este modelo debería incluir el proceso estocástico de los desempeños

$$G_j\left(t\right), \quad j=1,2,...,n$$

para cada elemento j del sistema, y la función estructura del sistema que produce el proceso estocástico correspondiente al desempeño del sistema completo

$$G\left(t\right)=\phi\left(G_1\left(t\right),...,G_n\left(t\right)\right).$$

En muchos casos prácticos, un modelo simple para MSS debería ser usado, este está basado en una distribución de probabilidad de los desempeños para todos los elementos del sistema en cualquier instante t durante el periodo de operación $[0,T]$ y la función de estrucutra del sistema

$$\begin{aligned} g_j, p_j\left(t\right);\ 1 \quad &\leq \quad j \leq n \\ &\qquad \phi\left(G_1\left(t\right),...,G_n\left(t\right)\right). \end{aligned}$$

Veamos el siguiente ejemplo (Lisnianski y Levitin 2003, página. 19):

1. Consideremos un sistema 2-3 MSS. Este sistema consiste de 3 componentes binarios, entonces, claramente tenemos

$$\begin{aligned} G_i(t) \quad &\in \quad \{g_{i1}, g_{i2}\} = \{0,1\}; \quad i=1,2,3. \\ g_{i1} \quad &= \quad \left\{\begin{matrix} 0; & si\ se\ tiene\ \ falla \\ 1; & si\ \ está\ \text{perfecto} \end{matrix}\right\}, \end{aligned}$$

luego el desempeño del sistema en el tiempo t, está dado como:

$$G(t)=\begin{cases} 0; \text{ si hay mas de un elemento con falla} \\ 1; \text{ si hay solamente un elemento con falla} \\ 2; \text{ si todos los elementos funcionan bien} \end{cases}.$$

Los valores de la función de estructura $G(t)=\phi\left(G_1(t),G_2(t),G_3(t)\right)$ para todos los estados posibles del sistema se muestran en la siguiente tabla.

Desempeño por elemento			Desemepeño del sistema entero
$G_1(t)$	$G_2(t)$	$G_3(t)$	$\phi(G_1(t), G_2(t), G_3(t))$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	2

Tabla 1 Desempeño del sistema mediante función estructura.

Se puede apreciar que el estado del sistema se encuentra determinado por los estados de sus componentes. Recordando que el estado por componente se cuentra representado por la v.a.$X_j(t)$; la cual indica el estado del $j - ésimo$ componente al tiempo t. El conjunto de estados correspondientes a cada uno de los n componentes forman el estado del sistema representado por el vector

$$(X_1, ..., X_n).$$

Así, la variable aleatoria $X(t)$ indica el estado entero del sistema al tiempo t. Sea $X(t) = \phi(X_1(t), ..., X_N(t))$; la función ϕ es la función de estructura mencionada anteriormente. Se representa la relación entre el vector de estados por componente y la variable o el estado del sistema. La confiabilidad de cada uno de los componentes, forman el vector de confiabilidad $\langle p_1(t), ..., p_n(t) \rangle$ del sistema. Este vector puede ser estimado o calculado en base a la función de estructura.

Dentro del análisis de confiabilidad de un sistema multi-estado, el sistema puede ser clasificado en dos tipos:

- MSS con componentes binarios. Para este caso cada componente puede realizar funciónes a una tasa: $G_j(t) = g_j$, si está trabajando y $G_j(t) = 0$ si falló.
- MSS con componentes multi-estado. Para este caso los componentes trabajan con distintas tasas hasta que hallan fallado completamente.

Utilizamos la función de estrucutra para determinar las K tasas de un sistema multi-estado. Tomamos el conjunto M que es el espacio de posibles valores del desempeño y tasas del sistema Cada estado del sistema $k = \{1, 2, ..., K\}$ corresponde a una

combinación única de estados de los n componentes del mismo, generando L^n con un número total de combinaciones: $K = \prod_{j=1}^{N} n_j$.

Supóngase que los elementos o componentes de un sistema son independientes, es decir, que su desempeño no está afectado por el de otro. Así, la probabilidad de cada combinación es igual al producto de las probabilidades de cada componente, así, para cierto grado de desempeño del sistema g_k, corresponde a una combinación de las tasas de cada componente $G_j(t)$, luego, la probabilidad p_k está dada por:

$$p_k(t) = \prod_{j=1}^{n} p_{j,i_j}(t),$$

mientras que el desempeño del sistema en el estado k es obtenido como:

$g_k(t) = \phi\left(g_{1,i_1}(t), ..., g_{N,i_N}(t)\right)$, i_j *es un estado específico para cada componente* j.

2.1.4. Tipos de MSS

De acuerdo con el modelo genérico, se pueden definir diferentes tipos de MSS describiendo el comportamiento estocástico de cada componente y definiendo la función de estructura del sistema. Se presentarán a continuación diferenctes modelos MSS que con muy comunmente ocupados en estudios de confiabilidad.

Estructura en Serie

Una conexión serie de los componentes de un sistema representa un caso donde una falla total o la falla de algún componente causa la falla del sistema completo. En un sistema MSS pueden considerarse dos tipos de estructura en serie:

- Transmisión: Un sistema utiliza la capacidad de sus componentes como una medida de su desempeño. La operación de este sistema está asociada al flujo que existe atraves de los componentes. (líneas de producción).

- Procesamiento: La medida de desempeño del sistema es caracterizanda mediante el tiempo operacional o la rapidez del proceso. La operación del sistema está asociada con tareas consecutivas ordenadas en una "linea"por componente. El tiempo total del sistema en operación es igual a la suma de los tiempos de operación de todos sus componentes $G(t) = \sum_{i=1}^{N} G_i(t)$. El estado de falla de un componente corresponde a una velocidad de procesamiento igual a cero, así el tiempo de operación del sistema se vuelve cero o infinito (comida rápida, un autolavado, etc.).

Ejemplos de la estructura en serie:

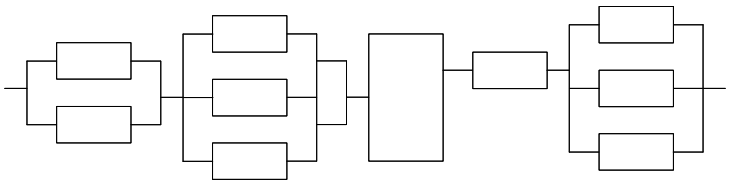

Figura. 1 MSS en serie tipo Transmisión/Procesamiento

Estructura Paralela

Se tiene una conexión paralela de los elementos, así el sistema falla si y solo si todos los componentes fallan. Los componentes en paralelo para el MSS permiten que alguna tarea puede ser realizada por cualquiera de los componentes paralelos $G(t) = \sum_{i=1}^{N} G_i(t)$. Se puede observar un sistema así en bancos, filas que despachan en un cine o en los supermercados. Puede presentarse el caso donde solamente un componente pueda trabajar a la vez.

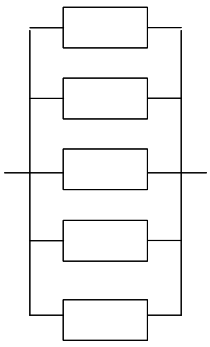

Figura. 2 Sistema en Paralelo (Líneas de prod. o CPU)

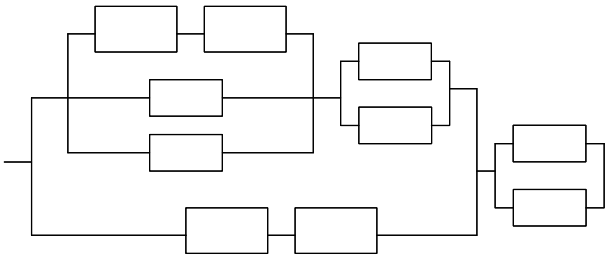

Figura. 3 Ejemplo MSS Serie-Paralelo.

K-N Estructura

Esta estructura correspondiente a algunos sistemas posee la condición de operación: al menos K elementos de los N se deben encontrar en condiciones operables. Este tipo

de sistemas corresponden a una estructura en serie, donde cada uno de los componentes forma una estructura en paralelo. En una generalización de MSS binario, MSS está en el estado j si al menos K_j componentes se encuentran en el estado $G_{j,i_j}(t)$.

Estructura Puente

El sistema se encuentra diseñado para asegurar el menor esfuerzo posible para los estados de funcionamiento y de falla. Como una estrategia de diseño, los componentes G_1 y G_2 poseen los mismos atributos operacionales, pero tienen diferentes niveles y probabilidades, lo mismo para los componentes G_3 G_4, un componente G_5, trabaja como interface entre los pares (G_1, G_4) y (G_2, G_3). Muchos sistemas de seguridad poseen una configuración como estas.

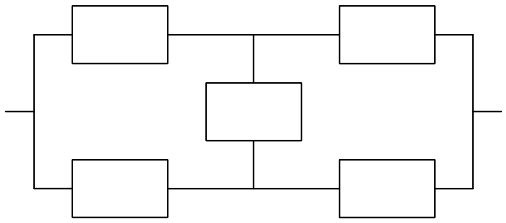

Figura 4 Ejemplo de MSS estructura Puente.

Sistemas con dos modos de falla

Estos sistemas consisten en modos de falla que pueden ocurrir en diferentes formas. Por ejemplo, sistemas con switch pueden fallar cuando se abre el sistema o cuando se cierra. Las fallas en modos abiertos o cerrados son eventos mutuamente excluyentes.

Weighted Voting Systems

Este sistema toma decisiones basadas en cada uno de los N independientes componentes. Cada componente $(j = 1, 2, ..., N)$, produce una decisión, la cual puede ser $G_j = 1$, (proposición aceptada), o $G_j = 0$ (proposición rechazada), puede no tomarse alguna decisión si se abstiene de dar una respuesta. Para realizar una decisión, el sistema incorpora todas las decisiones unitarias.Si al menos m aceptan, el sistema da una respuesta favorable. Un ejemplo de esto puede verse en las pruebas del VIH, cuando se está compuesto por mas de una sola prueba y se da una respuesta si al menos un número k da como resultado positivo.

Notemos que este tipo de MSS puede ser considerado un caso especial de la estructura $K - N$ con dos tipos de fallas. Dado que el desempeño del sistema puede variar, este tipo de sistemas pueden ser considerados como el caso más simple de MSS. Una

generalización de este tipo de sistemas es un sistema con pesos diferentes para cada elemento denotando la importancia relativa en el sistema.

Podemos observar distintos y muchos otros tipos de estructuras que envuelven un sistema MSS tales como:

- Sistemas MSS consecutivos,

- MSS de ventanas deslizables,

- Redes MSS,

- Modelos MSS mixtos,

Ahora mostraremos la relación entre los sistemas multi-estado y la confiabilidad de dichos sistemas.Índices de Confiabiliad de MSS

Para caracterizar numéricamente el comportamiento de MSS, se deben determinar ciertos índices de confiabilidad. Algunos de los siguientes índices están basados en el análisis del sistema es su tiempo de vida. Podemos considerar que existe cierta demanda que debe satisfacerse, entonces, en este caso, la relación entre el desempeño del sistema y la demanda representados por dos procesos estocásticos debe ser estudiado.

Cuando la distribución del sistema no depende del tiempo o del instante en el que se encuentre el mismo, el desempeño es representado como una variable aletoria que relaciona cada estado con cierta demanda correspondiente. Sea dos vectores w, q donde $w = \{w_1, ..., w_M\}$ el vector de los posibles niveles de demanda y $q = \{q_1, ..., q_M\}$ el vector de probabilidades estacionarias correspondiente a cada nivel de demanda, $q_j = P(W = w_j)$, $j = 1, ..., M$. Cuando uno considera que MSS evoluciona conforme pasa el tiempo o durante el periodo de vida del mismo $[0, T]$, las siguientes variables aleatórias deben ser consideradas:

1. Tiempo de falla (T_f) es el momento en el cual apartir del inicio de actividades del sistema este se encuentra en algún nivel innaceptable.

2. Tiempo entre fallas (T_b) es el tiempo entre dos transiciones consecutivas entre algún estado aceptable a uno innaceptable.

3. Número de fallas (N_T) ocasiones en que el sistema, hasta el tiempo T, posee niveles o estados innaceptables.

La variable aleatoria T_f es caracterizada por los siguientes índices:

- Función de Confiabilidad $R(t)$, es la probabilidada de que T_f sea mayor o igual a $t > 0$, donde en el estado inicial MSS se encuentra en un estado aceptable:

$$R(t) = P[T_f > t \mid F(G(0), W(0) \geq 0)].$$

Consideramos entonces dos procesos estocásticos $G(t)$ y $W(t)$. Asumiremos que el desempeño del sistema debe exceder el valor de la demanda: $F(G(t), W(t)) = G(t) - W(t)$.

- Tiempo medio a la falla ($MTTF$) es el tiempo medio al instante cuando el sistema completo se encuentra en algún estado o desempeño innaceptable por primera vez

$$E(T_f).$$

- La probabilidad de que el tiempo entre fallas sea mayor a t

$$P(T_b > t).$$

- El tiempo medio entre fallas ($MTBF$) :

$$E(T_b).$$

- La probabilidad de que N_T no sea mayor que cierto número s :

$$P(N_T \leq s).$$

- El número esperado de fallas del sistema en el intervalo $[0, T]$:

$$E(N_T).$$

- La disponibilidad instantánea $A(t)$ es la probabilidad de que MSS al instante $t > 0$ se encuentre en un estado o nivel aceptable:

$$A(t) = P[F(G(t), W(t)) \geq 0].$$

- La disponibilidad en el intervalo $[0, T]$ está definida como:

$$A_T = \frac{1}{T} \int_0^T 1\{F(G(t), W(t)) \geq 0\} dt,$$

así, A_T representa la porción de tiempo que MSS se encuentra en niveles aceptables. Para valores grandes de t el estado inicial del sistema prácticamente no tiene influencia alguna en la disponibilidad del mismo. Así, la disponibilidad estacionaria A para un nivel constante de demanda $W(t) = w$ puede ser determinada como:

$$A(w) = \sum_{k=1}^{K} p_k 1\{F(g_k, w) \geq 0\},$$

donde $p_k = \lim_{t \to \infty} p_k(t)$ es el vector de probabilidades estacionario de MSS correspondiente al estado k.

Cuando la demanda es variable, el periodo de operación de MSS hasta el tiempo T es particionado en M intervalos T_m $(1 \leq m \leq M)$ y un nivel de demanda constante w_m es asignado a cada intervalo m. En este caso el índice de disponibilidad puede ser escrito [*Billinton and Allan*, 1996] :

$$A(w,q) = \sum_{m=1}^{M} q_m \sum_{k=1}^{K} p_k 1\{F(g_k, w_m) \geq 0\},$$

y $w = \{w_1, ..., w_M\}$ es el vector de los posibles niveles de demanda y $q = \{q_1, ...q_M\}$ el vector de probabilidades estacionarias correspondientes a los niveles de demanda $(q_m = T_m/T)$.

Si se desean índices que caracterizen el desempeño promedio de MSS, se puede utilizar el desmpeño esperado. El valor medio del desempeño instantáneo al tiempo t está determinado como:

$$E_t = E(G(t)).$$

Si las probabilidades estacionarias $p_k = \lim_{t\to\infty} p_k(t)$ existen, el valor esperado estacionario toma la forma:

$$E_\infty = \sum_{k=1}^{K} p_k g_k.$$

El desempeño promedio en un intervalo de tiempo $[0, T]$ sería

$$E_T = \frac{1}{T}\int_0^T E_t \, dt.$$

Es importante de igual forma el medir cuánto faltó en el desempeño del sistema para satisfacer la demanda dada. En este caso, dado que consideramos que $F(G(t), W(t)) = G(t) - W(t)$, la desviación del desempeño instantánea es:

$$D(t) = \text{máx}\{W(t) - G(t), 0\}$$

y es llamada la deficiencia del desempeño al instante t. Dado que $D(t)$ es una variable aleatoria, pueden ser encontradas las siguientes medidas:

1. La probabilidad de que al instante t, $D(t)$ no exceda cierto nivel de demanda d

$$P(D(t) \leq d),$$

2. El valor medio de la deficiencia instantánea

$$D_t = E[D(t)].$$

Cuando MSS sigue probabilidades estacionarias y la demanda $W(t) = w$, la deficiencia no es una función del tiempo y puede ser obtenida como

$$D_\infty = \sum_{k=1}^{K} p_k \max\{w - g_k, 0\}.$$

En el caso de tener demanda variable, la deficiencia estacionaria es

$$D_\infty(w, q) = \sum_{m=1}^{M} \sum_{k=1}^{K} p_k q_m \max\{w_m - g_k, 0\}.$$

3. La deficiencia esperada en un intervalo $[0, T]$ está definida como:

$$D_T = \frac{1}{T} \int_0^T D_t \, dt$$

mientras que la deficiencia acumulada en dicho intervalo es:

$$L_T = D_T T.$$

Nótese que L_T es una variable aleatória.

4. La probabilidad de que L_T no exceda cierto nivel l es

$$P(L_T \leq l),$$

5. La cantidad esperada de producto no satisfecha a los clientes durante el intervalo $[0, T]$:

$$E(L_T).$$

El cálculo de muchos de estos índices o medidas en la mayoría de las ocasiones es difícil de realizar inclusive computacionalmente, sin embargo, se utilizará la técnica de la Función Generadora Universal para tal propósito.

2.1.5. La Función Generadora Universal (Universal Generating Function, UGF)

Fundamentos Matemáticos

Consideremos una variable aleatoria.X que toma valores en un conjunto finito, posee una función de distribución y sólamente puede ocupar un valor a la vez, por lo que todo evento que relacione los valores de la v.a. son mutuamente excluyentes. Como se sabe, existen funciones generadoras que determinan o permiten obtener información

acerca de la variable aleatoria de interés, por ejemplo la Función Generadora de Momentos $(f.g.m.)$ $E\left(e^{tX}\right) = M_X(t)$. Usando este concepto, si reemplazamos la función e^t por z, se obtiene la función

$$W(z) = E\left(z^X\right) = \sum_{i=0}^{k} z^{x_i} p_i.$$

Esta función es llamada la *transformación-z* de la v.a.X. Esta preserva algunas propiedades de la f.g.m. tales como que la primera derivada evaluada en $z = 1$ es el valor esperado de la v.a. De este modo, en general, la transformación-z de la suma de variables aleatorias independientes es el producto de cada transformación-z individual:

$$W\left(\sum_{i=1}^{n} X_i\right) = \prod_{i=1}^{n} W_{X_i}(z).$$

Ahora consideremos n variables aleatorias discretas $X_1, ..., X_n$ y asumimos que cada variable X_i posee una distribución representada por los vectores x_i y p_i. Si se desea evaluar la distribución de cualquier función arbitraria $f(X_1, ..., X_n)$ se tendrían que evaluar los vectores y y q correspondientes a los estados generados por dicha función y sus probabilidades.

Cada posible valor de la función f corresponde a una combinación de valores de sus argumentos. El número total de combinaciones es

$$K = \prod_{i=1}^{n} (k_i + 1)$$

donde k_i+1 es el número de las diferentes realizaciones de la variable aleatoria X_i. Dado que todas las variables son independientes, la probabilidad de cada combinación es el producto de las probabilidades de cada realización de los argumentos que conforman cierta combinación.

La probabilidad de que la $j-ésima$ combinación de las realizaciones de las variables puede ser obtenida como:

$$q_j = \prod_{i=1}^{n} p_{ij_i}$$

y el correspondiente valor de la función puede ser obtenido como

$$f_j = f(x_{1j_1}, ..., x_{nj_n}).$$

Cabe resaltar que algunas combinaciones diferentes pueden producir el mismo valor de la función, pero todas las combinaciones son mutuamente excluyentes. Así, la probabilidad de que una función tome algún valor es igual a la suma de las probabilidades de las combinaciones que producen dicho valor. Sea A_h el conjunto de combinaciones

que producen el valor f_h. Si el número total de realizaciones diferentes de la función $f(X_1, ..., X_n)$ es H, entonces la distribución de la función es:

$$y = (f_h : 1 \leq h \leq H), \quad q = \left[\sum_{(x_{1j_1},...,x_{nj_n}) \in A_h} \prod_{i=1}^{n} p_{ij_i} : 1 \leq h \leq H \right].$$

Definimos la *transformada-z* de cada variable aleatoria X_i que representa la distribución correspondiente a $(x_{i0}, ..., x_{ik_i})$, $(p_{i0}, ..., p_{ik_i})$ en la forma polinomial como anteriormente fue mencionado:

$$\sum_{j=0}^{k_j} p_{ij} z^{x_{ij}}.$$

Podemos ver que esta transformada es muy parecida a la función generadora de momentos de cualquier variable aleatoria, inclusive a la función generadora de probabilidades. Esta similitud también se conserva en las propiedades que presenta. Así el producto de los polonomios de ciertas transformadas-z correspondientes a las variables $X_1, ..., X_n$ determina la distribución de la suma de estas variables. Para mayor detalle acerca de funciones generadoras ver: Grimmet y Stirzaker, *Probability and Random Processes*, 2001.

De forma similar, se puede obtener la transformada-z que representa la distribución de cualquier función arbitraria f reemplazando el producto de los polinomios por un operador mas general sobre las transformadas-z de n variables aleatorias independientes representando la distribución correspondiente a cada una:

$$\underset{f}{\otimes} \left(\sum_{j_i=0}^{k_i} p_{ij_i} z^{x_{ij_i}} \right) = \sum_{j_1=0}^{k_1} \sum_{j_2=0}^{k_2} ... \sum_{j_n=0}^{k_n} \left(\prod_{i=0}^{n} p_{ij_i} z^{f\left(x_{1j_1},...,x_{nj_n}\right)} \right).$$

La técnica basada en el uso de la transformada-z y operadores compuestos es llamada la *Función Generadora Universal* (UGF). En este contexto la transformada-z de una variable aleatoria para la cual el operador $\underset{f}{\otimes}$ se encuentra definida como UGF. Dicha función para alguna variable X_i es denotada como $u_j(z)$ y la UGF para la variable $f(X_1, ..., X_n)$ como $U(z)$. De acuerdo a la notación:

$$U(z) = \underset{f}{\otimes} (u_1(z), u_2(z), ..., u_n(z)).$$

A pesar de el hecho de que la UGF tenga la forma de un polinomio, no es un polinomio pues los coeficientes y exponentes no son necesariamente variables escalares, pero pueden ser otros objetos matemáticos como vectores. Los operadores definidos pueden diferir de los operadores de productos polinomiales. En general, las UGF pueden ser utilizadas no solamente para representar la distribución de cierta variable. Sin embargo, en cualquier representación, los coeficientes se interpretan como las

probabilidades asociadas a cierta característica o algún objeto representado por los exponentes correspondientes a cada término.

Propiedades de composición de operadores

Las propiedades de la composición de operadores $\otimes_f$ depende estrictamente de las propiedades de la función f. Dado que la multiplicación de probabilidades bajo este operador es conmutativa y asociativa, el operador completo posee también estas propiedades si la función las posee. Si

$$f\left(X_1,...,X_n\right)=f\left[f\left(X_1,...,X_{n-1}\right),X_n\right],$$

entonces:

$$\begin{aligned} U\left(z\right) &= \underset{f}{\otimes}\left[u_1\left(z\right),u_2\left(z\right),...,u_n\left(z\right)\right] \\ &= \underset{f}{\otimes}\left[\underset{f}{\otimes}\left(u_1\left(z\right),...,u_{n-1}\left(z\right)\right),u_n\left(z\right)\right]. \end{aligned}$$

Si la función posee la propiedad asociativa

$$f\left(X_1,...,X_n\right)=f\left[f\left(X_1,...,X_j\right),f\left(X_{j+1},...,X_n\right)\right] \quad \forall j,$$

entonces el operador posee la propiedad

$$\underset{f}{\otimes}\left(u_1\left(z\right),u_2\left(z\right),...,u_n\left(z\right)\right)=\underset{f}{\otimes}\left[\underset{f}{\otimes}\left(u_1\left(z\right),...,u_j\left(z\right)\right),\underset{f}{\otimes}\left(u_{j+1}\left(z\right),...,u_n\left(z\right)\right)\right].$$

Si además la función posee la propiedad conmutativa $f\left(X_1,...,X_j,X_{j+1},...,X_n\right)=f\left(X_1,...,X_{j+1},X_j,...,X_n\right)$, el operador también posee dicha propiedad:

$$\underset{f}{\otimes}\left(u_1\left(z\right),...,u_j\left(z\right),u_{j+1}\left(z\right),...,u_n\left(z\right)\right)=\underset{f}{\otimes}\left(u_1\left(z\right),...,u_{j+1}\left(z\right),u_j\left(z\right),...,u_n\left(z\right)\right).$$

Puede también presentarse el caso que la función tome una forma recursiva

$$f\left[f_1\left(X_1,...,X_j\right),f_2\left(X_{j+1},...,X_h\right),...,f_m\left(X_l,...,X_n\right)\right],$$

entonces la función $U\left(z\right)$ puede ser obtenida recursivamente:

$$\underset{f}{\otimes}\left[\underset{f_1}{\otimes}\left(u_1\left(z\right),...,u_j\left(z\right)\right),\underset{f_2}{\otimes}\left(u_{j+1}\left(z\right),...,u_h\left(z\right)\right),...,\underset{f_m}{\otimes}\left(u_l\left(z\right),...,u_n\left(z\right)\right)\right].$$

Considerando una variable aleatoria X con distribución representada por su UGF $u_X\left(z\right)=\sum_{j=0}^{k}p_jz^{x_j}$. Para obtener la UGF que represente la distribución de $f\left(X,c\right)$ para una constante c, se puede aplicar el siguiente operador:

$$U\left(z\right)=u_X\left(z\right)\underset{f}{\otimes}c=\left(\sum_{j=0}^{k}p_jz^{x_j}\right)\underset{f}{\otimes}c=\sum_{j=0}^{k}p_jz^{f(x_j,c)}.$$

2.1.6. Confiabilidad de MSS y UGF

Dada la sección anterior y habiendo mostrado el uso y bondades de utilizar la función generadora universal en el estudio de la distribución de variables aleatorias, regresaremos al estudio de la confiabilidad de un sistema multi-estado en este caso, aplicando UGF para obtener algunos de los índices anteriormente mencionados, esto con la finalidad de determinar diversas características de un sistema para posteriormente mejorar las mismas.

Utilizando la función generadora universal, la función u representa una distribución de desempeño instantáneo para MSS, dicha función puede ser usada en la evaluación de la confiabilidad "instantánea"de un sistema, o para calcular un desempeño instantáneo medio, deficiencias medias instantáneas, etc. Teniendo la distribución del desempeño de MSS al tiempo $t \geq 0$ utilizando la representación de la función generadora universal

$$U(z,t) = \sum_{i=1}^{K} p_i(t)\, z^{g_i},$$

se puede obtener la disponibilidad al instante $t > 0$ para una demanda constante arbitraria w usando el operador δ_A :

$$A(t,w) = \delta_A[U(z,t), w] = \delta_A\left(\sum_{i=1}^{K} p_i(t)\, z^{g_i}, w\right) = \sum_{i=1}^{K} p_i(t)\, I_{(F(g_i,w)\geq 0)},$$

donde $F(g_i, w)$ es la función de aceptación.

El desempeño instantáneo esperado en el momento $t > 0$ puede ser obtenido dada $U(z)$ usando el operador δ_E :

$$E_t = \delta_E(U(z,t)) = \delta_E\left(\sum_{i=1}^{K} p_i(t)\, z^{g_i}\right) = \sum_{i=1}^{K} p_i(t)\, g_i.$$

Cuando el desempeño o comportamiento de MSS es representado por un valor escalar y $U(z)$ toma la forma de un polinomio genuino, el operador δ_E genera el valor de la primera derivada de $U(z,t)$ para $z = 1$;

$$\delta_E\left(\sum_{i=1}^{K} p_i(t)\, z^{g_i}\right) = \frac{dU(z,t)}{dz}(1) = \sum_{i=1}^{K} p_i(t)\, g_i.$$

Igualmente, el desempeño medio condicional de MSS (en el cual $F(g_i, w) \geq 0$) puede ser obtenido usando el operador δ_{CE} :

$$\begin{aligned} E^* &= \delta_{CE}(U(z,t)) = \delta_{CE}\left(\sum_{i=1}^{K} p_i(t)\, z^{g_i}\right) = \\ &\sum_{i=1}^{K} p_i(t)\, g_i I_{(F(g_i,w)\geq 0)} \;/\; \sum_{i=1}^{K} p_i(t)\, I_{(F(g_i,w)\geq 0)}. \end{aligned}$$

El desarrollo promedio esperado de MSS para un intervalo de tiempo $[0, T]$ está definido como:

$$E_T = \frac{1}{T}\int_0^T E(t)\,dt = \frac{1}{T}\sum_{i=1}^{K} g_i \int_0^T p_i(t)\,dt.$$

Para obtener ahora la deficiencia promedio instantánea para la función dada $U(z,t)$ y la demanda constante w, el operador δ_D debe ser usado:

$$D(t,w) = \delta_D(U(z),w) = \delta_D\left(\sum_{i=1}^{K} p_i(t)\, z^{g_i}, w\right) = \sum_{i=1}^{K} p_i(t) \times \max(w - g_i, 0).$$

Notar que si existen las probabilidades estacionarias correspondientes: $p_\infty = \lim_{t\longrightarrow\infty} p_i(t)$, cada una correspondiente a los estados del sistema, se puede determinar la disponibilidad A_∞ del sistema, la media estacionaria de desempeño E_∞ y la deficiencia media estacionaria D_∞ reemplazando $p_i(t)$ por $p_{i\infty}$.

Cabe destacar que UGF no considera las características de las transiciones del sistema entre diferentes estados, y por lo tanto, no puede ser usada para evaluar algunos índices como el tiempo medio para que falla o el número esperado de fallas.

Veamos ahora un ejemplo en el cual podamos ver los conceptos ya definidos:

Example 1 *(Lisnianski y Levitin, página 162). Considerar un sistema que tiene tres diferentes grados de desempeño:*

$$g_1 = 0, g_2 = 20 \quad y \quad g_3 = 40$$

cuyas probabilidades correspondientes a cada g_i son:

$$\begin{aligned} p_1(t) &= 0{,}043\exp(-23{,}478t) - 0{,}106\exp(-9{,}552t) + 0{,}063, \\ p_2(t) &= -0{,}289\exp(-23{,}478t) - 0{,}0246\exp(-9{,}552t) + 0{,}313, \\ p_3(t) &= 0{,}246\exp(-23{,}478t) + 0{,}13\exp(-9{,}552t) + 0{,}624. \\ p_1(t) + p_2(t) + p_3(t) &= 1 \end{aligned}$$

(este sistema tiene 3-estados por elemento con fallas mínimas y cuyas reparaciones poseen una intensidad de transición $\lambda_{2,1} = 2{,}02/años$, $\lambda_{3,2} = 7{,}01/años$; $\mu_{1,2} = 10/años$, $\mu_{2,3} = 14/años$).

La distribución del desempeño del sistema en cualquier instante $t > 0$:

$$g = \{0, 20, 40\}, \quad p = \{p_1(t), p_2(t), p_3(t)\},$$

puede ser representada por la siguiente función generadora universal:

$$U(z,t) = p_1(t)\, z^0 + p_2(t)\, z^{20} + p_3(t)\, z^{40}.$$

El sistema MMS falla si el desempeño decae tal que no pueda satisfacer una demanda $w = 15$. Entonces, la función de aceptación toma la forma:

$$F(g_i, w) = g_i - 15.$$

Así, podemos obtener las siguientes expresiones que nos proporcionan información acerca de la confiabilidad del sistema:

1. Disponibilidad Instantánea:

$$\begin{aligned} A(t) &= \delta_A(U(z,t),15) = \delta_A\left(\sum_{i=1}^{3} p_i(t)\, z^{g_i}, 15\right) = \sum_{i=1}^{3} p_i(t)\, I_{[F(g_i,15)\geq 0]} \\ &= p_2(t) + p_3(t) = -0{,}043\exp(-23{,}478t) + 0{,}106\exp(-9{,}552t) + 0{,}937. \end{aligned}$$

2. El desempeño instantáneo esperado:

$$\begin{aligned} E_t &= \delta_E(U(z)) = \delta_E\left(\sum_{i=1}^{3} p_i(t)\, z^{g_i}\right) = \sum_{i=1}^{3} p_i(t)\, g_i = 20p_2(t) + 40p_3(t) \\ &= 4{,}047\exp(-23{,}478t) + 4{,}730\exp(-9{,}552t) + 31{,}223. \end{aligned}$$

3. El desempeño promedio esperado durante un intervalo de tiempo $[0, T]$:

$$\begin{aligned} E_T &= \frac{1}{T}\int_0^T E(t)\,dt = \frac{1}{T}\sum_{i=1}^{K} g_i \int_0^T p_i(t)\,dt = \frac{1}{T}\left[20\int_0^T p_2(t)\,dt + 40\int_0^T p_3(t)\,dt\right] = \\ &\quad \frac{1}{T}\int_0^T (4{,}047\exp(-23{,}478t) + 4{,}730\exp(-9{,}552t) + 31{,}223)\,dt \\ &= \frac{1}{T}(0{,}667 - 0{,}172\exp(-23{,}478T) - 0{,}495\exp(-9{,}552T)) + 31{,}223. \end{aligned}$$

   Podemos pensar si $T = 0{,}5$, $E_T = 32{,}55$, o si $T = 1$ año, $E_T = 31{,}89$.

4. La deficiencia media instantánea:

$$\begin{aligned} D(t) &= \delta_D(U(z,t),15) = \delta_D\left(\sum_{i=1}^{3} p_i(t)\, z^{g_i}, 15\right) = \sum_{i=1}^{3} p_i(t)\,\text{máx}(15 - g_i, 0) = \\ 15p_1(t) &= 0{,}650\exp(-23{,}478t) - 1{,}597\exp(-9{,}552t) + 0{,}947. \end{aligned}$$

5. Podemos ver facilmente que las probabilidades estacionarias son:

$$\begin{aligned} p_1 &= \lim_{t\to\infty} p_1(t) = 0{,}063; \\ p_2 &= \lim_{t\to\infty} p_2(t) = 0{,}313; \\ p_3 &= \lim_{t\to\infty} p_3(t) = 0{,}624. \end{aligned}$$

6. Por otra parte, se pueden obtener A_∞; E_∞ y D_∞ como:

$$\begin{aligned} A_\infty &= p_2 + p_3 = 0{,}937; \\ E_\infty &= 20p_2 + 40p_3 = 31{,}223 \\ D_\infty &= 15p_1 = 0{,}947. \end{aligned}$$

El siguiente ejemplo muestra de modo general el uso de la función de estructura para un sistema multi-estado, el cómo se obtienen los estados de cada uno así como las probabilidades correspondientes a cada combinación, y la obtención de la función $U(z)$:

Example 2 *(Lisnianski y Levitin, página 156)Considérese un sistema de transmisión de gas que consiste de dos elementos (pipas). El grado de desempeño de la linea de pipas es definido por la capacidad de transmisión, esta puede tomar diferentes valores discretos dependiendo del estado de control del equipo. El primer elemento posee tres posibles estados* ($k_1 = 3$) *con grados de desempeño* $g_{13} = 1$, $g_{12} = 0{,}7$ *y* $g_{11} = 0$, *con probabilidades* $p_{13} = 0{,}8$, $p_{12} = 0{,}15$, $p_{11} = 0{,}05$. *El segundo elemento posee dos posibles estados* ($k_2 = 2$) *con grados de desempeño* $g_{21} = 0$ *y* $g_{22} = 1$ *y probabilidades correspondientes* $p_{22} = 0{,}9$, $p_{21} = 0{,}1$.

El número total de combinaciones de estados en el sistema es $K = k_1 k_2 = 6$. Cada combinación define un único estado del sistema, veamos la siguiente tabla:

Estados del sistema	estados de los elementos	desempeño del sistema $\phi(G_1, G_2) = G_1 + G_2$	probabilidad del edo. del sistema
1	$\{g_{13}, g_{22}\} = \{1, 1\}$	$g_1 = g_{13} + g_{22} = 2$	$p_{13}p_{22} = 0{,}720$
2	$\{g_{13}, g_{21}\} = \{1, 0\}$	$g_2 = g_{13} + g_{21} = 1$	$p_{13}p_{21} = 0{,}080$
3	$\{g_{12}, g_{22}\} = \{0{,}7, 1\}$	$g_3 = g_{12} + g_{22} = 1{,}7$	$p_{12}p_{22} = 0{,}135$
4	$\{g_{12}, g_{21}\} = \{0{,}7, 0\}$	$g_4 = g_{12} + g_{21} = ,7$	$p_{12}p_{21} = 0{,}015$
5	$\{g_{11}, g_{22}\} = \{0, 1\}$	$g_5 = g_{11} + g_{22} = 1$	$p_{11}p_{22} = 0{,}045$
6	$\{g_{11}, g_{21}\} = \{0, 0\}$	$g_6 = g_{11} + g_{21} = 0$	$p_{11}p_{21} = 0{,}005$

Tabla 2 Estados y probabilidades correspondientes al sistema en paralelo.

Asumimos que las líneas de las pipas están conectadas en paralelo y la capacidad total de transmisión del sistema está determinada como la suma de las capacidades de cada pipa. La función de estructura es $\phi(G_1, G_2) = G_1 + G_2$. Aquí $G_1 \in \{g_{11}, g_{12}, g_{13}\}$ y $G_2 \in \{g_{21}, g_{22}\}$. Si la línea de pipas estuviera conectada en serie, el total de la capacidad de transmisión está restringida por la pipa con menor capacidad de transmisión;

así la función de estructura sería $\phi(G_1, G_2) = \text{mín}\{G_1, G_2\}$:

Estados del sistema	estados de los elementos	desempeño del sistema $\phi(G_1, G_2) = \text{mín}\{G_1, G_2\}$	probabilidad del edo. del sistema
1	$\{g_{13}, g_{22}\} = \{1, 1\}$	$g_1 = \text{mín}\{g_{13}, g_{22}\} = 1$	$p_{13}p_{22} = 0{,}720$
2	$\{g_{13}, g_{21}\} = \{1, 0\}$	$g_2 = \text{mín}\{g_{13}, g_{21}\} = 0$	$p_{13}p_{21} = 0{,}080$
3	$\{g_{12}, g_{22}\} = \{0{,}7, 1\}$	$g_3 = \text{mín}\{g_{12}, g_{22}\} = {,}7$	$p_{12}p_{22} = 0{,}135$
4	$\{g_{12}, g_{21}\} = \{0{,}7, 0\}$	$g_4 = \text{mín}\{g_{12}, g_{21}\} = 0$	$p_{12}p_{21} = 0{,}015$
5	$\{g_{11}, g_{22}\} = \{0, 1\}$	$g_5 = \text{mín}\{g_{11}, g_{22}\} = 0$	$p_{11}p_{22} = 0{,}045$
6	$\{g_{11}, g_{21}\} = \{0, 0\}$	$g_6 = \text{mín}\{g_{11}, g_{21}\} = 0$	$p_{11}p_{21} = 0{,}005$

Tabla 3 Estados y probabilidades correspondientes al sistema en serie.

Estas tablas muestran la distribución del desempeño del sistema basadas en la distribución del desempeño de cada elemento y la función de estructura. La misma distribución del sistema puede ser obtenida usando la transformación-z tal que refleje el producto de polinomios.

Sea el polinomio

$$\sum_{i_j=1}^{k_j} p_{ji_j} z^{g_{ji_j}},$$

que representa la distribución de desempeño $\left\{g_{ji_j}, ..., g_{jk_j}\right\}, \left\{p_{ji_j}(t), ..., p_{jk_j}(t)\right\}$ del elemento j. Este polinomio es la expresión correspondiente a la función generadora correspondiente a cada elemento. El producto de los polinomios correspondientes a los elementos $1, ..., n$ toma la forma

$$\prod_{j=1}^{n}\left[\sum_{i_j=1}^{k_j} p_{ji_j} z^{g_{ji_j}}\right] = \sum_{i_1=1}^{k_1}\sum_{i_2=1}^{k_2}...\sum_{i_n=1}^{k_n}\left(\prod_{j=1}^{n} p_{ji_j} z^{g_{ji_j}}\right).$$

En este procedimiento podemos verificar que la función de estructura es exactamente la misma. En el caso de las líneas de gas en paralelo, podemos representar las distribuciones de desempeño de los elementos usando estos polimonios:

$$\begin{aligned} u_1(z) &= p_{11}z^{g_{11}} + p_{12}z^{g_{12}} + p_{13}z^{g_{13}} = 0{,}8z^1 + {,}15z^{,7} + {,}05z^0, \\ u_2(z) &= p_{21}z^{g_{21}} + p_{22}z^{g_{22}} = 0{,}9z^1 + {,}1z^0, \end{aligned}$$

así, se obtiene la representación de la distribución de desempeño del sistema entero como el producto de los polinomios:

$$\begin{aligned} U(z) &= \left(p_{11}z^{g_{11}} + p_{12}z^{g_{12}} + p_{13}z^{g_{13}}\right)\left(p_{21}z^{g_{21}} + p_{22}z^{g_{22}}\right) \\ &= 0{,}72z^2 + {,}08z^1 + {,}135z^{1{,}7} + 0{,}015z^{,7} + {,}045z^1 + {,}005z^0. \end{aligned}$$

Notemos que esta distribución pudo haber sido obtenida directamente de la tabla mediante la forma

$$\sum_{i=1}^{K} p_i z^{g_i}.$$

Si deseamos obtener la distribución de desempeño de MSS con una función de estructura arbitraria ϕ utilizando el polinomio anterior, elegimos el operador $\underset{f}{\otimes}$ sobre las transformaciones-z de los n elementos del sistema:

$$\underset{f}{\otimes}\left(\sum_{i_1=1}^{k_1} p_{1i_1} z^{g_{1i_1}}, ..., \sum_{i_n=1}^{k_n} p_{ni_n} z^{g_{ni_n}}\right) = \sum_{i_1=1}^{k_1}\sum_{i_2=1}^{k_2}...\sum_{i_n=1}^{k_n}\left(\prod_{j=1}^{n} p_{ji_j} z^{\phi\left(g_{1i_1},...,g_{ni_n}\right)}\right).$$

Aplicar el operador $\underset{f}{\otimes}$ con la función estructura $\phi\left(G_1, G_2\right) = \text{mín}\left\{G_1, G_2\right\}$ sobre las funciones generadoras individuales de cada pipa en el ejemplo, se puede obtener la distribución de desempeño del sistema completo en el caso de ser en serie:

$$\begin{aligned} U\left(z\right) &= \underset{f}{\otimes}\left(p_{11}z^{g_{11}} + p_{12}z^{g_{12}} + p_{13}z^{g_{13}}, p_{21}z^{g_{21}} + p_{22}z^{g_{22}}\right) \\ &= p_{13}p_{22}z^{\text{mín}\{g_{13},g_{22}\}} + p_{13}p_{21}z^{\text{mín}\{g_{13},g_{21}\}} + p_{12}p_{22}z^{\text{mín}\{g_{12},g_{22}\}} + \\ &\quad p_{12}p_{21}z^{\text{mín}\{g_{12},g_{21}\}} + p_{11}p_{22}z^{\text{mín}\{g_{11},g_{22}\}} + p_{11}p_{21}z^{\text{mín}\{g_{11},g_{21}\}}. \end{aligned}$$

Para los cinco parámetros, el polinomio toma la forma:

$$\begin{aligned} U\left(z\right) &= \underset{f}{\otimes}\left(0{,}8z^1 + {,}15z^{,7} + {,}05z^0, 0{,}9z^1 + {,}1z^0\right) \\ &= 0{,}72z^{\text{mín}\{1,1\}} + {,}08z^{\text{mín}\{1,0\}} + {,}135z^{\text{mín}\{1,,7\}} + 0{,}015z^{\text{mín}\{,7,0\}} + \\ &\quad {,}045z^{\text{mín}\{0,1\}} + {,}005z^{\text{mín}\{0,0\}}, \\ &= 0{,}72z^1 + {,}08z^0 + {,}135z^{,7} + 0{,}015z^0 + {,}045z^0 + {,}005z^0. \end{aligned}$$

Como puede observarse, el estudio de la confiabilidad de un sistema multi-estado es accesible mediante el uso de la función generadora universal y de los diferentes operadores que proporcionan los índices del sistema. A continuación se presentará el uso de UGF en el estudio de sistemas en serie-paralelo y su relación con el método de diagramas de bloques pues en el desarrollo del trabajo posterior se utilizará dicho método en conjunto con la función generadora universal en MSS serie-paralelo.

2.1.7. Función Generadora Universal y MSS serie-paralelo

Método de Diagramas de Bloque

En algunas ocasiones la función de estructura de MSS no posee algunas de las propiedades vistas anteriormente (asociativa), pero puede ser representada como una

composición de funciones estructura que sí posean dichas propiedades. Estas funciones corresponden a algunos subconjuntos de elementos de MSS y permiten mediante sus correspondientes UGF obtener la función correspondiente al sistema total. El método que distingue los subconjuntos de MSS y permite reemplazarlos con elementos simples equivalentes está basado en una representación gráfica de la estructura del sistema y es llamado el método de diagrama de bloques de confiabilidad.

UGF de MSS en serie En sistemas multi-estado de transmisión de procesos en los cuales el desempeño del mismo está definido como su capacidad o productividad, la capacidad total de un subsistema que contiene n elementos conectados en serie es igual a la capacidad del elemento que sea el çuello de botella", es decir, el elemento con menos desempeño. Entonces, la función de estructura de tal subsistema toma la forma

$$\phi_s\left(G_1,...,G_n\right)=\prod_{i=1}^{n}G_i;$$

para el caso cuando se tengan elementos binarios, la función de estructura del sistema puede escribirse como

$$\phi_s\left(G_1,...,G_n\right)=\text{mín}\left\{G_1,...,G_n\right\}.$$

Cuando se tiene un sistema que procesa diferentes tareas, el desempeño está definido por la rapidez del procesamiento o el tiempo de operación de cada elemento del sistema que posee su propio desempeño, así el tiempo total para completar ciertas tareas se encuentra restringido. Generalmente el tiempo total que toma el sistema para completar cierta tarea es mayor que el tiempo necesario para realizarlo. Esto puede ser causado por el deterioro de los elementos del mismo que causan atrasos o que el desempeño sea insatisfactorio.

Consideremos un sistema que consiste de n elementos con sus fallas totales conectadas en serie. Cada elemento j tiene solo dos estados: operación con una tasa de desempeño g_{j1} y de falla con tasa de desempeño 0. La probabilidad del estado operacional es p_{j1}. La UGF de un elemento es

$$u_j\left(z\right)=\left(1-p_{j1}\right)z^0+p_{j1}z^{gj1},\quad j=1,...,n.$$

Para encontrar la UGF del sistema completo, apliquemos el operador $\Omega_{\phi s}$, el cual, da por resultado la función del sistema

$$U\left(z\right)=\Omega_{\phi s}\left[u_1\left(z\right),...,u_n\left(z\right)\right]=\left(1-\prod_{j=1}^{n}p_{j1}\right)z^0+\prod_{j=1}^{n}p_{j1}z^{\text{mín}\{g_{11},...,g_{n1}\}}.$$

UGF de sistemas con elementos en paralelo En los sistemas MSS de transmisión de procesos en los cuales se puede dispersar dicho proceso y ser transferido a diferentes elementos en paralelo simultáneamente, la capacidad total de un subsistema que contiene n elementos conectados en paralelo es igual a la suma de las capacidades de cada elemento. Así, la función de estructura para la cual un subsistema toma la forma

$$\phi_p\left(G_1, ..., G_n\right) = \sum_{j=1}^{n} G_j.$$

Considerando un sistema que posee n elementos en paralelo, se puede ver que si el sistema comienza a realizar ciertos procesos con los componentes en las mejores condiciones, los procesos estarán completos cuando sean terminados por al menos uno de los elementos. El tiempo total del proceso del sistema se encuentra definido por el tiempo mínimo de proceso de sus elementos y la rapidez total del sistema está definida por la velocidad máxima de los elementos.

Tomando el ejemplo de Levitin y Lisnianski (página. 170) podemos ver el cómo funciona el operador Ω_{ϕ_p} que nos proporciona la UGF para el sistema en paralelo. Suponer que se tienen dos elementos que forman un sistema en paralelo. Los elementos poseen tasas de desempeño g_{11} y g_{21} con probabilidades p_{11} y p_{21} respectivamente. La UGF del sistema entero está definida por el operador Ω_{ϕ_p} como:

$$\begin{aligned} U\left(z\right) &= \Omega_{\phi_p}\left(u_1\left(z\right), u_2\left(z\right)\right) \\ &= \Omega_{\phi_p}\left(\left(1-p_{11}\right)z^0 + p_{11}z^{g_{11}}, \left(1-p_{21}\right)z^0 + p_{21}z^{g_{21}}\right), \end{aligned}$$

cuya función de estructura queda determinada por

$$U\left(z\right) = \left(1-p_{21}\right)\left(1-p_{11}\right)z^0 + p_{11}\left(1-p_{21}\right)z^{g_{11}} + \left(1-p_{11}\right)p_{21}z^{g_{21}} + p_{11}p_{21}z^{g_{11}+g_{21}}.$$

UGF de sistemas serie-paralelo. La función de estructura de systemas serie-paralelo que poseen algún tipo de estructura compleja pueden ser siempre representados como una composición de las funciones estructura de subsistemas estadísticamente independientes que contienen solamente elementos conectados en serie o en paralelo. Así, para obtener UGF de un sistema serie-paralelo se deben aplicar operadores recursivos para conseguir las UGF de cada subsistema en serie o paralelo según su estructura. El siguiente algoritmo describe el procedimiento para dicho propósito (Lisnianski y Levitin 2003, página.171):

1. Encontrar los subsitemas pertenecientes a MSS que sean puramente en serie y paralelo.

2. Obtener las UGF de estos subsistemas usando el operador Ω_ϕ correspondiente.

3. Reemplazar cada subsistema por elementos simples que posean una UGF representado la estructura que poseen.

4. Si MSS posee más elementos, regresar al paso 1.

A continuación se presenta un ejemplo de un sistema en serie-paralelo para mostrar el cómo obtener la correspondiente UGF en el caso del diagrama de bloques de confiabilidad. Este ejemplo es también de la referencia anteriormente citada. Consideremos el sistema mostrado en la Figura 5.A. Primero puede obtenerse la UGF correspondiente a los elementos representados por $u_2(z)$, $u_3(z)$ y $u_4(z)$. Se calcula la función $U_1(z) = \Omega_{\phi_s}(u_2(z), u_3(z), u_4(z))$ y se reemplaza estos tres elementos por uno simple con función $U_1(z)$, se obtiene un sistema con estructura mostrada en la Figura 5.B. Este sistema contiene simplemente un subsistema en paralelo que consiste de elementos cuyas funciones son $U_1(z)$ y $u_5(z)$.

Se reemplazan los elementos $U_1(z)$ y $u_5(z)$ por otro cuya función sea $U_2(z) = \Omega_{\phi_p}(U_1(z), u_5(z))$ resultando una estructura como la Figura 5.C. Así, queda una estructura en serie de tres elementos que pueden ser reemplazados por un elemento simple con UGF $U_3(z) = \Omega_{\phi_s}(u_1(z), U_2(z), u_6(z))$, dicha estructura se ve en la Figura 5.D. Como se puede apreciar, resulta un sistema que consta de dos elementos conectados en paralelo. La UGF de esta última estructura representa la distribución del sistema entero inicial como $U(z) = \Omega_{\phi_p}(U_3(z), u_7(z))$.

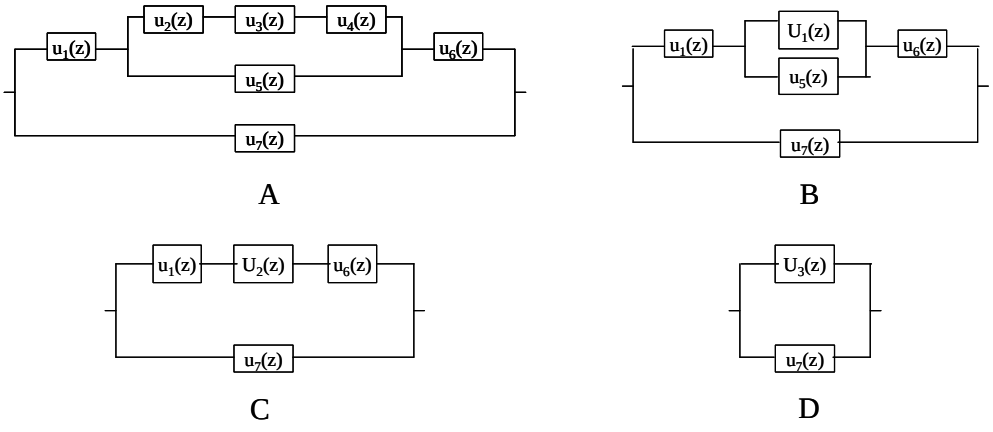

Figura. 5. Ejemplo de determinación recursiva del sistema MSS con UGF

De igual forma, la UGF correspondiente a MSS puede ser obtenida de forma recursiva. Si se tiene un sistema con n elementos donde subsistemas conectados en serie están formados por elementos estructurados en paralelo, algunos de estos a su vez son subsistemas en serie, luego se tiene

$$\phi(G_1, G_2, ..., G_n) = \phi_s\left[\phi_p(G_1, ..., \phi_s(G_j, G_{j+1}, ...), G_i, G_{i+1}), ...G_n\right].$$

Si el sistema tuviera elementos conectados en paralelo y algunos de ellos subsistemas en serie, la composición para obtener la UGF se vería como

$$\phi\left(G_1, G_2, ..., G_n\right) = \phi_p\left[\phi_s\left(G_1, ..., \phi_s\left(G_j, G_{j+1}, ...\right), G_i, G_{i+1}\right), ...G_n\right].$$

Este procedimiento recursivo para obtener UGF correspondiente a MSS no suele se mas conveniente que el método directo pero, este es más importante pues permite reducir el tiempo computacional necesario para procesar el algoritmo. Usando el procedimiento directo uno tiene que evaluar la función de estructura del sistema para cada combinación de valores de las variables aleatorias $G_1, G_2, ..., G_n$ $\left(\prod_{j=1}^{n} k_j \ veces\right)$. Utilizando el algoritmo recursivo uno puede tomar ventaje del hecho de que algún subsistema puede poseer las mismas tasas de desempeño en diferentes estados, los cuales hacen a estos estados indistinguibles y reducen por tanto el número total de términos en las correspondientes funciónes-u.Optimización de Confiabilidad de MSS

Ya estuados los conceptos básicos que requiere un sistema multi-estado para su comprensión y descripción, así como la relación que se guarda con los índices de confiabilidad, la función generadora universal y los distintos métodos que relaciona la UGF con la función de estructura, es de interés el estudio de la mejora de la confiabilidad de un sistema pues no solamente se desea que este último trabaje hasta que su tiempo de vida termine o hasta que sus componentes lo permitan. Puede pensarse en asuntos de costos, tanto en industrias como en empresas el tema de reducción de costos o ahorros potenciales es de suma importancia y relacionado con un sistema que se posea será relevante el poder mantener trabajando un sistema el mayor tiempo posible con el menor costo de mantenimiento a un cierto nivel de confiabilidad.

Es así, como el tema de la mejora de la confiabilidad es de suma importancia en varios tipos de sistemas. Sin embargo, puede presentarse una situación en la cual los recursos necesarios para realizar dicha mejora son limitados; es decir, se cuentan con restricciones económicas o tecnológicas. Así, se han identificado dos enfoques respecto a este problema de optimización, el primero es el buscar maximizar la confiabilidad del sistema sujeta a diversas restricciones; mientrads que el segundo es el minimizar los recursos necesarios para alcanzar un nivel de confiabilidad requerido. Levitin señala que hay por lo menos cuatro métodos generales para mejorar la confiabilidad de un sistema:

- Método de redundancias
- Un ajuste óptimo de los parámetros del sistema, un re-ordenamiento óptimo de los elementos del sistema
- Mejora de la confiabilidad y/o el desempeño de los elementos del sistema

- Una combinación de los puntos antes mencionados.

El aplicar a un MSS alguno de estos métodos afecta basicamente dos propiedades del sistema: su configuración y el desempeño o distribución de sus elementos. En un problema de optimización de estrucutra simple de MSS, cada subsistema puede contener solamente elementos idénticos. Este problema es relevante cuando se desea proveer de un mantenimiento a cierto sistema o prevenir compras frecuentes de elementos para reparar el sistema. Un problema característico de optimización es el elegir qué componentes reemplazar por otros con mejores características o qué elementos se deben adherir al sistema.

En algunos sistemas la importancia de la confiabilidad de diferentes elementos puede depender de forma significativa de su ubicación en el sistema completo. Cuando los elementos son intercambiables, el arreglo que posean puede mejorar considerablemente la confiabilidad del sistema. Cuando el número de componentes es igual al número de elementos, se tiene un problema de secuenciación óptima. En general se puede apreciar el cómo un problema de optimización debe considerar la distribución de recursos limitados entre los elementos de MSS para alcanzar la mayor confiabilidad posible o para alcanzar un nivel deseado con mínimos recursos.

Un ejemplo de este tipo de problema es la asignación de pruebas de crecimiento de confiabilidad. En este problema los modelos son usados para estimar la confiabilidad de los elementos del sistema como una función del tiempo de prueba; el tiempo de prueba para cada elemento debe ser determinado para maximizar la confiabilidad del sistema entero cuando los recursos de prueba son limitados. Otro ejemplo es la optimización de normas o políticas de mantenimiento para todo elemento de un sistema. Teniendo estimaciones sobre la influencia de diferentes acciones de mantenimiento preventivo o correctivo en la confiabilidad de los elementos del sistema, se puede evaluar la importancia en el sistema completo de dichas acciones. Así, una política óptima de mantenimiento debe responder preguntas como: ¿Qué elementos deben ser primordiales en el mantenimiento? o ¿Cuál debe ser la intensidad de estas actividades?.

Dado que las actividades de mantenimiento y la incorporación de redundancias mejoran la confiabilidad de MSS, la pregunta inmediata es cuál es mejor.

2.1.8. Algoritmos genéticos para optimizar

Se han utilizado varios métodos de optimización para reolver problemas de mejora de la confiabilidad de sistemas, dichos métodos han sido heurísticas o algoritmos exactos basados en programación dinámica y no-lineal. Muchos de estos métodos están orientados a resolver un cierto tipo de problemas, esto les impide ser facilmente adaptados a resolver otro tipo de problemas. Recientemente, muchos trabajos sobre

optimización de confiabilidad están basados en metaheurísticas. Estas metaheurísticas dependen en muchas ocasiones de la naturaleza del problema a resolver pudiendo así ser aplicadas a una amplia gama de problemas de optimización. Estas están basadas tanto en la inteligencia artificial como en la programación matemática. Una de las ventajas más importantes es que no requieren información acerca de las funciones objetivo, pues se basan en los puntos probables del espacio de soluciones.

Todas las metaheurísticas utilizan el concepto de aleatorización cuando se tiene una búsqueda aunado al conocimiento previo adquirido, así, dirigen las actuales y las posteriores. Cuatro son las metaheurísticas comunmente usadas en estos casos:

- búsqueda tabú
- algoritmos genéticos
- colonía de hormigas
- simulated annealing.

A continuación se hablará acerca de las metaheurísticas y aquella que será ocupada en este estudio, así como sus elementos, su base matemática y su forma de aplicación.

Las Metaheuristicas

Antecedentes Existen algoritmos capaces de resolver problemas combinatorios que proporcionan una solución "óptima" lo suficientemente razonable en un tiempo considerable. Por otra parte existen problemas cuyo grado de complejidad es tal, que los algoritmos antes mencionados no logran obtener resultados satisfactorios en tiempo y forma adecuados. Es por esto que se han logrado desarrollar procesos de búsqueda de resultados que satisfagan las condiciones propuestas por el problema llamados 'heurísticas'.

El origen etimológico de esta palabra es el mismo que la de la palabra eureka, sin embargo 'heurística' aparece más en las categorías gramaticales. Se usaba como sustantivo para aludir a la ciencia del descubrimiento y al mismo tiempo como adjetivo para señalar estrategias o reglas heurísticas. Una de las características de las heurísticas es la creatividad y el pensamiento lateral que poseen para poder resolver de otro modo problemas con un grado mayor de complejidad. El concepto fue introducido por el matemático George Pólya. Así, las heurísticas podrían ser consideradas como métodos de búsqueda que satisfagan las condiciones de algún problema a resolver para encontrar soluciones que maximicen o minimicen una función objetivo, en otras palabras, que la optimicen.

Cabe aclarar que no siempre las condiciones del problema permanecen constantes durante la resolución del mismo, este es un problema que los algoritmos sencillos no pueden considerar, pues puede que aumente o disminuyan las posibilidades de búsqueda . Para este estudio, se tratará el problema de optimización de políticas de mantenimiento en sistemas multi-estado. Es claro que la naturaleza de dicho problema es complicada de tratar mediante algoritmos que manejan cierto grado de simplicidad en sus procesos.

Problemas como este poseen naturaleza combinatoria cuyo grado de complejidad es mayor dado que poseen restricciones adicionales o mayor información determinística que las heurísticas no pueden considerar y menos resolver, para la resolución de estos se desarrollaron las Metaheurísticas.

Definición de Metaheurísticas Este término fue introducido al estudio de la optimización por primera vez en 1986 por Fred Glover. Una metaheurística se refiere a una gran estrategia que guía y modifica otras heurísticas para producir mejores soluciones que las generadas por heurísticas y otros métodos o algoritmos. Poseen tres características generales: 1) el uso de memoria adaptable, 2) el tipo de vecindario para la exploración de soluciones, y 3) el número de soluciones halladas y tomadas por buenas después de una iteración a otra.

Las metaheurísticas tienen la capacidad de guardar información acerca de los procedimientos que se están realizando para buscar el valor que satisfaga la función objetivo, específicamente guarda los óptimos locales para poder hacer comparaciones en las diferentes etapas del proceso que se está realizando. Esta búsqueda se realiza de modo indefinido, hasta un determinado número de iteraciones definidas por el usuario, pues de otro modo, el proceso podría 'nunca terminar'. Cabe aclarar que la mayoría de las metaheurísticas están diseñadas para resolver problemas de optimización combinatoria, cuya naturaleza puede ser polinomial o no (problemas NP o NP-hard respectivamente).

En este estudio se utilizará específicamente la metaheurística llamana Ant Colony Optimization y su desarrollo como una herramienta estocástica, así como su aplicación en la solución de problemas de optimización combinatoria.

Colonia de Hormigas (Ant Colony Optimization, ACO)

Antecedentes Esta metaheurística está basada en el comportamiento natural de la colonias de hormigas, en el cómo desarrollan un sistema de búsqueda de alimento y de selección de vías rápidas para el mismo fin. Las hormigas poseen una capacidad para establecer el camino más corto desde su hormiguera hasta la fuente de alimento

y de modo similar al revés, es este comportamiento el que motiva a la creación de esta metaheurística para hallar caminos de búsqueda rápida y eficiente.

Cuando una hormiga se desplaza desde su nido hasta la fuente de alimento, va dejando tras ella un rastro de una sustancia llamada "feromona". Las feromonas son un sistema de comunicación química entre hormigas o animales de la misma especie, cuyo objetivo es la transmisión de la información a través de señales odoríferas acerca del estado de los mismos. Esta sustancia entonces, sirve como guía a otras hormigas para seguir el rastro del camino que lleva al alimento. De este modo, las siguientes hormigas reconocen el rastro de feromona que fue depositado por las otras y deciden seguirlo con una 'probabilidad' alta de elección de este camino sobre otros que no posean feromonas. Así, el camino más frecuentado y elegido una mayor cantidad de veces tendrá una mayor probabilidad de elección que los otros debido a la cantidad de feromonas depositadas en el. Hay que aclarar que los caminos que dejan de ser recorridos sufren de un proceso llamado 'evaporación de feromonas', lo cual, les quita de algún modo cierta probabilidad de elección contra los que son recorridos más frecuentemente.

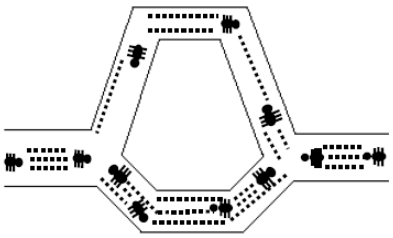

Figura 7.excursión 1

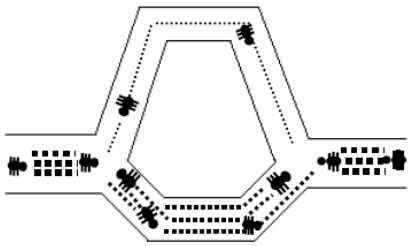

Figura 8.excursión $n - ésima$

La metaheurística que se estudiará será Ant System, que es un algoritmo particular de Ant Colony Optimization. Esta idea fue tomada y transformada por Marco Dorigo, cuyo objetivo principal era el encontrar soluciones a problemas difíciles de tratar mediante la construcción de agentes que intercambian información por medio de una matriz de feromonas.

Problemas de optimización combinatoria La optimización combinatoria es una rama de la optimización en matemáticas aplicadas y en ciencias de la computación, relacionada a la investigación de operaciones, teoría de algoritmos, inteligencia artificial e ingeniería de software. Los algoritmos de optimización combinatoria resuelven instancias de problemas que se creen ser difíciles en general, explorando el espacio de soluciones para estas instancias, logrando reducir el tamaño efectivo del espacio, y explorando el espacio de búsqueda eficientemente.

Cada modelo matemático presenta:

- Una serie de decisiones cuantificables, las variables de decisión.
- Una función objetivo que mide la efectividad de cada sistema de decisiones.
- Una serie de restricciones sobre las decisiones a tomar que delimitan el conjunto de soluciones posibles

De manera formal, los problemas de optimización combinatoria pueden ser definidor por tres características $(S,\ f,\ \Omega)$; donde S es el conjunto de posibles soluciones, f es la función objetivo que asignará un valor $f(s)$ a cada candidato a ser una solución $s \in S$, y Ω es un conjunto de necesidades a cumplirse o de restricciones. El objetivo es encontrar una solución que funja como un óptimo global $s\prime$. Generalmente para poder resolver este tipo de problemas se requiere de la construcción de un programa computacional por el peso de la búsqueda que llega a tener el problema y el número de iteraciones que se necesitan para llegar a una buena solución.

En la optimización de problemas combinatorios discretos, la mayoría de las ocasiones se trata con problemas del tipo NP-completos, esto quiere decir que no es posible resolverlos en un tiempo polinomial. Este tipo de problemas de optimización son aquellos que buscan maximizar o minimizar una función asociada para la cual existen soluciones viables numerables o no. Un ejemplo muy comunmente utilizado es el problema del Agente Viajero (TSP, Taveling Salesman Problem). El objetivo principal de este problema es el hallar una ruta óptima de costo mínimo dentro de un circuito Hamiltoniano, es decir dentro de una gráfica de nodos y pesos; ciudades y distancias a recorrer comenzando desde un “nodo” y terminando en el mismo.

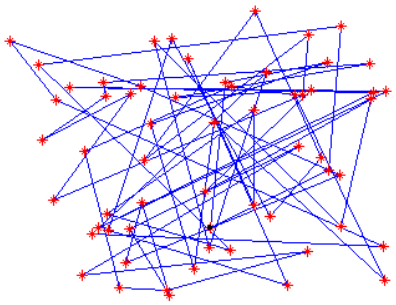

Figura 9 representación del problema del agente viajero

Del modelo biológico al modelo matemático Para pasar del modelo biológico al matemático, consideraremos el caso particular de Ant System (AS) como se había mencionado en las secciones anteriores. Para dicho propósito, consideraremos la construcción de "gráficas aleatorias"para poder representar soluciones factibles como una caminata dentro de la misma. Dichas gráficas son llamadas "gráficas de costrucción".

Definition 3 *Dado un problema de optimización combinatoria, se tiene una grafica de construcción $C = (V, A)$ junto con una función Φ con las siguientes propiedades:*

- En C existe un único nodo llamado nodo inicial.

- Sea W el conjunto de todas las caminatas o rutas w en C que satisfacen:

w comienza en el nodo inicial

w contiene todos los nodos de C al menos una vez

El último nodo de w no tiene sucesor.

Se asume que el problema combinatorio a tratar será minimizado puesto que la función objetivo representará costos. Cabe aclarar que existen muchos modos de construir gráficas, sin embargo, el desarrollo del algoritmo tendrá un impacto en la misma construcción. Si las tres condiciones anteriores no son satisfechas, entonces se considerará un camino no factible.

Ahora, se describirá "Graph based in Ant System"(Gutjahr, 2000) como una extensión del mismo Ant System con la finalidad de definir los conceptos que el modelo matemático utilizar en el algoritmo, esto servirá para posteriormente definir el algoritmo a utilizar. Los elementos son los siguientes:

1. Una gráfica de construción $C = (V, A)$ que contendrá todas los nodos posibles a visitar.

2. Un conjunto $A_1, A_2, ..., A_s$ de agentes ("hormigas") donde cada uno construye una caminata aleatoria con ciertas restricciones eligiendo los nodos a visitar en base a una matriz que contiene probabilidades de transición. El nodo $i \rightarrow j$, representa el ir del punto i al j. Cada una de las caminatas o caminos de las hormigas es construido separadamente y de modo secuencial. El periodo de tiempo en el que una hormiga construye un camino es llamado *ciclo* y definimos L_k como la distancia total recorrida en el camino de la hormiga k.

3. Las probabilidades de transición de los movimientos aleatorios de las hormigas dentro de cada ciclo se encuentran en una matríz P que es de transición.

   Sea $u = (u_0, ..., u_{t-1})$ el camino parcial construido por una hormiga antes del $t - ésimo$ paso en un ciclo m. Así, u_0 indica el nodo inicial, u_1 el siguiente nodo a u_0, etc. Decimos que $l \subset u$ si l está contenido en u. Si A es el conjunto de arcos que une los nodos en la gráfica de construcción, entonces la forma general de las probabilidades de transición están dadas por:

$$p_{kl}(m, u) = \begin{cases} \frac{[\tau_{kl}(u)]^{\alpha}[\eta_{kl}(u)]^{\beta}}{\left(\sum_{(k,r)}[\tau_{kr}(u)]^{\alpha}\left[\eta_{k,r}(u)\right]^{\beta}\right)}; & \text{si } l \in u \\ 0; & \text{si } l \notin u \end{cases}$$

Así, $p_{kl}(m,u)$ denota la probabilidad de que una hormiga teniendo un camino parcial u en el ciclo m, se mueva del nodo k a l. Aquí $\alpha, \beta > 0$ son parámetros que nos indican el porcentaje de importancia relativa que se le asigna al proceso de toma de decisión en cuanto a las cantidades del rastro de feromona y a la información heurística respectivamente que posee el modelo. Al inicio de cada ciclo, cada hormiga es posicionada en una ciudad inicial para poder construir su propia gráfica. Una vez terminado el ciclo, cada camino costruido por las hormigas constituye una solución para el problema.

4. La cantidad de Feromonas τ_{kl} constituyen una matríz τ. La cantidad de feromonas τ_{kl} es la correspondiente al arco que une los nodos $k-l$. Los valores de la matriz de feromonas cambian ciclo a ciclo, así las feromonas del ciclo m son descritas como: $\tau_{kl}(m)$. Para inicializar la matríz de feromonas se considera: $\tau_{kl}(0) = \frac{1}{M}$, donde M es el número de arcos de un nodo (k,l), así, al final de cada ciclo, la actualización de feromonas se lleva a cabo de la siguiente manera:

   - Evaporación: a cada valor de la matríz de feromonas se le aplica un procedimiento llamado .$^{\text{Ev}}$aporación de Feromonas "dado por $\tau_{kl} = (1-\rho)\tau_{kl}$. Aquí ρ es una tasa de evaporación, $(1-\rho)$ indica el porcentaje de feromonas que quedan en un arco; $0 < \rho \leq 1$.
   - Depósito: Una vez llevada a cabo la evaporación, se realiza el depósito de feromonas en cada uno de los arcos de la mejor solución hallada $\widehat{x}$ en cada ciclo de la siguiente forma: $\tau_{kl} = \tau_{kl} + \frac{\rho}{L(\widehat{x})}$, donde $L(\widehat{x})$ es la longitud total del mejor camino encontrado, siempre y cuando los nodos k, l estén secuenciados y pertenezcan a la mejor solución encontrada.

   Esta regla de depósito y evaporación de feromonas puede ser interpretada de la siguiente manera: Si un camino no es visitado, las cantidades de feromonas correspondientes a sus nodos son evaporadas, así, este camino será menos probable de ser elegido en algún ciclo. De modo contrario, aquellos caminos más frecuentados sufren de depósito de feromonas, estos serán posteriormente los que puedan ser elegidos con mayor probabilidad.

5 Información heurística $\eta_{k,r}(u)$ entre todos los nodos k, r que pertenezcan al conjunto C. Dado que se puede contar con información del tipo: distancias entre dichos, tiempos de ir de i a j, costo de ir de i a j, la cantidad $\eta_{k,r}(u)$ nos porporciona información acerca de cuan atractivo es el secuenciar después de un nodo k el nodo r. Esta cantidad está dada por el recíproco de dicha distancia, dicho tiempo o costo; es decir, para el caso donde se tengan las distancias en una matriz D, donde cada d_{kr} es la distancia entre nodos, $\eta_{k,r}(u) = \frac{1}{d_{kr}}$.

Esta cantidad puede ser obtenida con la ayuda de heurísticas "Greedy"(o voraces HG) que construyen de manera rápida"la información que proporciona atractivo a ciertos caminos o no, sin embargo, en muchos casos se supone que la informacion HG ya está dada.

Una vez descritos los elementos necesarios para realizar un camino para cada hormiga, podemos generalizar lo que el algoritmo hace: Una vez posicionadas las hormigas en un nodo inicial, se elige el siguiente según la matríz de probabilidades P y cierta regla de decisión. Una vez terminado cada tour, se actualiza la matríz de feromonas, se elige la mejor solución encontrada y es guardada. Este proceso se repite un gran número de veces para poder encontrar la solución óptima a la cual convergen cada una de las hormigas.

Bajo estas consideraciones, la metaheurística ACO puede ser vista como una serie de procesos estocásticos que trabajan en un espacio de soluciones de un problema de optimización combinatoria. Aunado a esto, se puede observar cómo las hormigas artificiales son independientes heurísticas estocásticas constructivas entre si que producen mejores soluciones para un problema de optimización combinatoria usando caminos donde dispersan feromonas.

Estas han sido algunas generalidades de ACO como metaheurística, sin embargo, el propósito de este trabajo no es abordar los problemas que se resuelven como el algoritmo determinístico, sino el cómo se puede asegurar una convergencia a la solución óptima. Para ello se trabajará con la extensión de ACO a problemas estocásticos, es decir, que contenga variables aleatorias, y al mismo tiempo, el algoritmo .^{es}tocástico". Antes de ello, veremos cómo la construcción de la solución para el GBAS puede ser visto como un proceso de Markov, esto es que la probabilidad de transición solamente depende el estado anterior. De modo similar al caso determinístico, se tiene una alta probabilidad de converger a la solución óptima. Se ha pensado que ACO es una metaheurística que ayuda en gran forma a resolver este tipo de problemas por tres razones:

- Trabaja con una memoria lo que le proporciona robusticidad. (mas que a Simulated Annealing simple y Tabú Search)
- También puede aplicarse a problemas con muchas restricciones y un espacio de soluciones reducido.
- Algunas características de otras heurísticas pueden ser adheridas para mejorar el desempeño de ACO.

Esto le proporciona a ACO ventajas importantes para resolver problemas con restricciones en optimización combinatoria.

ACO como Cadena de Markov El sistema hormiga (Dorigo y Massienzzo 1991) como ya hemos visto, es un nuevo tipo de metaheuristica inspirado en la recolección de comida de una colonia de hormigas real, la cual ha demostrado trabajar bien para un serie de estudios experimentales. Diversas modificaciones del sistema hormiga (AS por sus siglas en ingles) han sido aplicadas a diferentes tipos de problemas de optimización combinatoria discreta teniendo excelentes resultados.

Para la investigación de ACO, se introduce, como ya ha sido definido, una estructura formal base-hormiga llamado "Sistema Hormiga basado en grafos"(GBAS,-Graph-Based-Ant-System). Es de interés en este estudio el mostrar la convergencia de ACO a la solución óptima de un problema. Para esto se utiliza que el procedimiento descrito anteriormente induce a un proceso de Markov, es decir, el procedimiento para encontrar una solución óptima mediante ACO-GBAS forma un proceso de Markov en tiempo discreto (Gutjahr, 2000). Los estados de este proceso de Markov están dados por las ternas:

$$(\underline{\tau}(n), \underline{w}(n), f^*(n)) \qquad (n = 1, 2, ...)$$

donde:

- $\underline{\tau}(n)$ es el vector de valores de feromonas $\tau_{kl}(n)$ para todos los arcos (k, l) durante la iteración o ciclo n.
- $\underline{w}(n)$ es el vector de caminatas $w^{(s)}(n)$ $(s = 1, 2, ..., S)$ para las hormigas $A_1, A_2, ..., A_S$ en la iteración n.
- $f^*(n)$ es el mejor valor de la función objetivo encontrada para cualquier agente de las iteraciones $1, 2, ..., n-1$, (esto es, el valor f^* que controla la actualización de las feromonas al final de la iteración n) para $m = 1$ se define $f^*(1) = \infty$.

La terna anterior forma una cadena de Markov, pues para que la propiedad de Markov se satisfaga la distribución de los estados en la iteración n $(\underline{\tau}(n), \underline{w}(n), f^*(n))$, solo debe depender de los estados de la iteración $n-1$ $(\underline{\tau}(n-1), \underline{w}(n-1), f^*(n-1))$. En ACO-GBAS efectivamente se cumple, ya que los estados de transición estan dados como sigue

- $\underline{\tau}(n)$ es el resultado deterministico de $\underline{\tau}(n-1), \underline{w}(n-1)$ y $f^*(n-1)$ según la regla de actualización de los valores de las feromonas.
- La distribución de $\underline{w}(n)$ solo depende de $\underline{\tau}(n)$ ya que los valores $\eta_{kl}(u)$ son deterministicos pues se conocen cuando se tiene una caminata parcial u, la cual está perfectamente determinada en la iteración $n-1$. Por lo anterior $\underline{\tau}(n)$ solo depende de la iteración $n-1$.

- $f^*(n)$ es el resultado deterministico de $\underline{w}(n-1)$ y $f^*(n-1)$.

Cabe aclarar que las probabiliades $p_{kl}(n,u)$ definidas para ACO solo dependen de $\underline{\tau}(n)$, así como $p_{kl}(n,u)$ es función de los estados del proceso de Markov definido por las ternas $(\underline{\tau}(n), \underline{w}(n), f^*(n))$, $p_{kl}(n,u)$ induce probabilidades de transición para la cadena de Markov definida para GBAS. Estas probabilidades no están dadas explícitamente, es decir, no se tiene una expresión matemática para estas ya que en la mayoría de las ocasiones es imposible generarlas. Esto puede pensarse debido a que el proceso de Markov que define el GBAS hace transiciones globalizadas, es decir, las transiciones entre las ternas $(\underline{\tau}(n), \underline{w}(n), f^*(n))$, son las correspondientes en cada iteración del algoritmo.

Para aclarar las ideas descritas, se puede observar lo siguiente: si definimos $X_n = (\underline{\tau}(n), \underline{w}(n), f^*(n))$, para el proceso de Markov $(X_n)_{n\geq 1}$, las probabilidades de trancisión cumplen:

$$\mathbb{P}(X_n = x_n \mid X_{n-1} =_{x-1}, ..., X_0 = x_0) = \mathbb{P}(X_n = x_n \mid X_{n-1} = x_{n-1}) = p_{n-1,n}$$

donde: $p_{n-1,n}$ es la probabilidad correspondiente a dicha transición del proceso de Markov visto como cadena.

Se tiene entonces que $f^*(n)$ se encuentra definido por la caminata $\underline{w}(n-1)$ y $f^*(n-1)$, donde $\underline{w}(n-1)$ está dado por $\underline{\tau}(n-1)$ que a su vez depende de los valores de la transición $n-2$. Esto nos lleva a que la probabilidad de pasar de un estado X_{n-1} a uno X_n depende de varios factores que en cada paso de la iteración van cambiando, por lo cual las probabilidades de transición también lo hacen. Con esto se tiene que estas probabiliades $P_{n-1,n}$ no poseen una forma explicita, pudiéndo pensarse en que si el número de ciclos no es finito, la matriz de transición de este proceso de Markov puede ser infinita.

Lo aclarado anteriormente nos lleva a concluir que este proceso de Markov es no homogeneo, ya que las probabilidades de transición dependen de la iteración n en la que se encuentre.

Problema del Agente Viajero Generalmente la introducción del cómo se maneja o desarrolla ACO se realiza mediante el tratamiento del problema del agente viajero. A pesar de que el problema a tratar es diferente al mencionado anteriormente, se introducirá esta metaheurística con los conceptos del TSP.

Se tienen un conjuto de N ciudades que deben ser visitadas por un viajero, saliendo de una ciudad s y volviendo al final del recorrido a la misma cuidad inicial. Una vez elegida una ciudad inicial, se debe crear una ruta .ºptima"que recorra todas las ciudades dentro del conjuto, es decir, crear un circuito Hamiltoniano donde se van

conectando las ciudades visitadas creando una ruta. Se desea minimizar costos, recorrer todas las ciudades en el menor tiempo o distancia posible. En este caso, se trabajará con distancias,se buscará reducir la distancia total del recorrido. Se deben considerar todas las rutas posibles ($N!$) y de estas, elegir la "óptima".

Así, definimos $f(x, \omega)$ como la distancia total generada al recorrer el camino x. Ahora, definiremos en forma general el funcionamiento del algoritmo para posteriormente demostrar que se está trabajando con la Cadena de Markov que tiene convergencia a la solución óptima.

Extensión a S-ACO (Stochastic Ant Colony Optimization) para problemas combinatorios estocásticos

Dado que estamos tratando con un proceso de Markov, estudiaremos la extensión de ACO para problemas combinatorios estocásticos. La extención del algoritmo GBAS, mostrado en la sección anterior, es el algoritmo llamado S-ACO en el caso de problemas de optimización estocastica (1). S-ACO deja el procedimiento básico en gran parte sin cambios, pero modifica el subprocedimiento de la actualización de las feromonas mediante la introducción de una prueba estocastica, donde si la solución guardada como la mejor encontrada, se sigue considerando como la mejor.

En el pseudocodigo descrito a continuación, escribiremos $\tau_{kl}(n)$ para referirnos a τ_{kl} y denotar la dependencia que se tiene en cuanto al número de la iteración correspondiente, de la misma manera se entiende $p_{kl}(n)$.

La viabilidad de un arco (k, l) en una caminata parcial u con terminación en el nodo k se define de la misma manera que en la sección anterior. El arco (k, l) es factible si el nodo l aun no está contenido en u, y ninguna de las restricciones adicionales espeficadas para l son no factibles antes de que u haya sido hecho.

Pseudocódigo: **procedure** S-ACO{
inicializar la matríz de feromonas
for round $n = 1, 2, ..., \{$
for hormiga $\sigma = 1, 2, ..., s\{$
sea k, La posición actual de la hormiga, igual al nodo inicial en $\mathcal{C}$;
sea u, La caminata actual de la hormiga, igual a una lista vacia;
While (exista un nodo factible (k, l) para la caminata u de la hormiga σ) {
selccionar un nodo sucesor l con probabilidad p_{kl}, donde

$$p_{kl} = \begin{cases} 0 & \text{si } (k,l) \text{ es no factible} \\ \frac{\tau_{kl}(u)\eta_{kl}(u)}{\left(\sum_{(k,r)} \tau_{kr}(u)\eta_{k,r}(u)\right)} & \text{si } (k,l) \text{ es factible} \end{cases}$$

la suma se da sobre todos los (k, r) factibles;
sea $k = l$ y adicionar l a u;
sea $x_\sigma = u$;
}
de $\{x_1, ..., x_s\}$ seleccionar un camino x;
hacer S-pheromone-update (x, n);
}
}
Procedure S-pheromone-update (x, n) {
calcular el estimado

$$\widetilde{F}(n) = \frac{1}{n} \sum_{\nu=1}^{N_n} f(x, \omega_\nu)$$

para cada hormiga en una iteración; elegir la mejor $\widetilde{F}_\sigma(n)$, para posteriormente:

set $\widehat{x} = x$ la mejor solución encontrada;
}
hacer, con $L(\widehat{x})$ denotando la magnitud del camino $\widehat{x}$,

$$\tau_{kl}(n+1) := \begin{cases} (1-\rho)\tau_{kl}(n) + \frac{\rho}{L(\widehat{x})}, & \text{si } (k,l) \in \widehat{x} \\ (1-\rho)\tau_{kl}(n), & \text{en otro caso} \end{cases}$$

}

Convergencia de S-ACO hacia la solución óptima

Finalmente se debe enunciar el resultado más importante para justicar el uso de ACO: La convergencia de S-ACO a la solución óptima global. Para que el algoritmo presentado tenga validez y sustento teórico, se utiliza una justificación teorica que menciona que la mejor solución producida por el algoritmo converge con probabilidad 1 a la solución optima global.

Para poder tener la convergencia dicha solución óptima, se necesitan las siguientes condiciones (Gutjahr, 1999):

(i) La solución optima x^* es unica.

(ii) La función observada $f(x, \omega_\nu)$ de un escenario aleatorio ω_ν puede descomponerse en el valor esperado y un termino de error de la siguiente manera:

$$f(x, \omega_\nu) = f(x) + \epsilon_{x\nu},$$

donde $f(x) = \mathbb{E}(f(x, \omega))$, y $\epsilon_{x\nu}$ se distribuye $N\left(0, \sigma^2(x)\right)$; notando que todos los errores son independientes.

(iii) Los valores de atracción satisfacen:

$$\eta_{kl}(u) > 0, \text{ para todo tramo } u \text{ de } x^* \text{ y para todo } (k,l) \text{ en } x^*$$

(iv) Existe un limite inferior de feromonas tal que:

$$\tau_{\text{mín}}(n) = \frac{c_n}{\log(n+1)} \quad \text{con} \quad \lim_{n\to\infty} c_n > 0.$$

(v) El tamaño de la muestra N_n se incrementa linealmente, i.e., $N_n > C \cdot n$ para alguna constante $C > 0$.

(Gutjahr 1999)Si las condiciones $(i) - (v)$ se satisfacen, entonces para el mejor camino encontrado $\widehat{x}(n)$ en la iteración n, para los valores de las feromonas $\tau_{kl}(n)$ y para la probabilidad $\mathbb{P}_\sigma(n)$ de alguna hormiga σ que atraviesa x^* el camino óptimo; las siguientes afirmaciones se cumplen

$$\lim_{n\to\infty} \widehat{x}(n) = x^* \textit{ casi seguramente}$$

$$\lim_{n\to\infty} \tau_{kl}(n) = \begin{cases} 1/L(x^*) & si \ (k,l) \in x^* \\ 0 & en\ otro\ caso \end{cases} \quad \textit{casi seguramente}$$

$$\lim_{n\to\infty} \mathbb{P}_\sigma(n) = 1$$

Capítulo 3

Metodología

Ya habiendo definido las herramientas teóricas, numéricas y los modelos matemáticos adecuados que son necesarios en el desarrollo de este trabajo, se describirá la metodología a seguir así como los algoritmos que son necesarios implementar para llegar a la resolución de cada uno de los problemas a tratar. En términos generales se estudiarán los pasos a seguir para obtener una solución óptima.

Recordando un poco el contexto en el cual se desarrolla este trabajo, se tratarán dos tipos de problemas de optimización de políticas de mantenimiento, uno de ellos sin considerar el tiempo en el que cada mantenimiento debe ser hecho y el otro considerando la distribución del tiempo de falla de cada elemento y del sistema entero. En primer lugar se desarrollarán los elementos necesarios respecto a un sistema multi-estado y los índices de confiabilidad que se necestian evaluar para posteriormente aplicar la metaheurística ACO y obtener la mejor polítca de mantenimiento.En este capítulo se definirá el concepto de política de mantenimiento, las restricciones que se necesitan cumplir y el cómo se considerará una solución mejor a otra.

3.1. Problema de optimización en políticas de mantenimiento

En términos generales tenemos un sistema multi-estado cuyos elementos se encuentran conectados en serie y algunos de ellos son subsistemas que poseen una estructura en paralelo. Dado el tiempo de vida o de trabajo de dicho sistema, es evidente que la confiabilidad del mismo se va reduciendo a lo largo del tiempo pues pueden existir causas naturales tales como el desgaste normal de los elementos o alguna falla que se haya originado por cuestiones agenas al desarrollo del sistema, pero haya causado que la confiabilidad del sistema entero decayera de un modo más rápido.

Se desea entonces que el sistema provea cierto nivel de demanda durante todo el

tiempo de vida manteniendo un cierto nivel mínimo de confiabilidad. Estas van a ser las dos condiciones más importantes a respetar en la búsqueda de soluciones pues el buscar una política de mantenimiento óptima permitirá satisfacer estas restricciones. Al hablar de una política de mantenimiento nos referimos a la secuencia de mantenimientos que deben aplicarse a ciertos elementos en un cierto momento. Dicha secuencia de mantenimientos debe incluir también el concepto de "prevención". Es claro que el papel que juegan los costos es importante, pues si un sistema ha fallado en sus totalidad, el realizar reparaciones y perder tiempo de producción es altamente costos para cualquier empresa, pues puede pensarse que se tratan de gastos no planeados. Es así como el planear una secuencia de mantenimientos preventivos resulta más eficiente y barato, pues permite que el sistema sigua trabajando o produciendo lo necesario para satisfacer la demanda dada.

En este caso, el mantener un cierto nivel de confiabilidad es importante para el desempeño del sistema, así cumplir con la demanda y evitar que haya fallas frecuentemente. Como se había mencionado anteriormente, se resolverán y optimizarán dos diferentes tipos de problemas. Se especificará la metodología a utilizar para cada uno.

3.1.1. Optimización de mantenimientos MSS

Los problemas de optimización de mantenimiento pertenecen a la clase de problemas de optimización de asignación y confiabilidad (Kuo y Prasad 2000). Estos problemas tratan de mejorar el sistema con el manejo de los diferentes elementos para mejorar la confiabilidad del sistema completo, es decir, incrementando la confiabilidad de los elementos del sistema llevando a cabo diferentes acciones de mantenimiento permite mejorar la confiabilidad del sistema entero. Por otra parte, incrementa el costo de mantenimientos el realizar acciones preventivas. En los sistemas MSS el papel de cada elemento en la mejora de la confiabilidad dependen de su distribución de desempeño y su lugar en el sistema, así es claro que la distribución óptima de los recursos limitados del mantenimiento es un problema combinatorio complicado.

A continuación se presentará la metodología para tratar el Problema 1 el cual es tomado de Lisnianski y Levitin sección 6.3.1 (2003).

3.1.2. Problema 1: Política Óptima de reemplazos cíclicos

Cuando un sistema posee elementos con diferentes tasas de falla que se incrementan con el paso del tiempo, un mantenimiento preventivo es muy conveniente haciendo los elementos tan buenos como nuevos o reemplazándolos, es decir mediante remplazos o mantenimientos preventivos (PR preventive replacement), los cuales son llamados mantenimientos preventivos perfectos. Un mantenimiento alternativo puede llevarse a

cabo cuando ha ocurrido una falla y el sistema debe serguir trabajando con el menor costo posible, es decir con una reparación mínima (MR) o con un mantenimiento correctivo. Este mantenimiento permite a los elementos del sistema continuar trabajando sin afectar sus tasas de falla; suele ser menos costoso.

Mantenimientos preventivos para elementos que posean una tasa de falla relativamente alta, reduce la probabilidad de falla, sin embargo, puede causar gastos significativos, en especial en sistemas con una alta tasa de reemplazos. Puede vislumbrarse entonces, que el crear una política de mantenimiento con acciones PR y MR podría generar una solución óptima para problemas con diferentes criterios.

Uno de los pasos más importantes en la implementación de un mantenimiento óptimo, es la obtención de las distribuciones de los tiempos de vida por elemento. Se está interesado en poder conocer el número esperado de fallas de elementos en cualquier intervalo de tiempo, así, dada la distribución del tiempo de vida, dicho valor esperado puede ser obtenido con modelos matemáticos o datos experimentales. En el modelo MR, cuya duración es relativamente menor comparada con el tiempo entre fallas, el número esperado de fallas es igual al número esperado de reparaciones para cualquier intervalo de tiempo. Así, es posible obtener la función de renovación para cada elemento, es decir, el número esperado de reparaciones en un intervalo de tiempo $(0, t)$. Este número esperado de fallas/reparaciones $f(t_j)$ puede ser estimado para diferentes intervalos de tiempo $(0, t_j)$ entre acciones PR consecutivas.

Para el Problema 1 se determinará una secuencia óptima de acciones PR cíclicas para un sistema multi-estado con una configuración serie-paralelo y elementos con dos estados. Cada elemento de este sistema está caracterizado por su tasa de desempeño y su función de renovación. Los tiempos y costos de los dos tipos de mantenimiento se encuentran disponibles para cada elemento. El objetivo es proveer de un cierto nivel de confiabilidad o disponibilidad al sistema mediante la suma de costos mínimos de mantenimientos.

Este método supone independencia entre los reemplazos y las actividades de reparación de diferentes elementos del sistema. Otra suposición importante es que los tiempos de reparaciones y de reemplazos son mucho más pequeños que el tiempo entre fallas. En este caso, la probabilidad de que un reemplazo (PR) y una reparación (MR) coincidan es nula. En sistemas con demanda cíclica variable, el PR puede ser desarrollado en periodos de baja demanda aún si las reparaciones de algún elemento no son terminadas. Por ejemplo, en sistemas de generación de poder, algunos elementos importantes deben ser reparados por la noche cuando la demanda es mucho menor. Aquí las acciones de mantenimiento deben ser consideradas independientes.

Formulación del problema

Consideramos un sistema que consta de n elementos. Para cada elemento j $(1 \leq j \leq n)$ se tiene una tasa de desempeño g_{j1}. Los tiempos y costos correspondientes a los mantenimientos preventivos y correctivos se encuentran dados, así como la función de renovación que representa el número esperado de elementos que fallan o son reparados en el intervalo de tiempo $(0, t)$. Para cualquier intervalo de reemplazos t_j, para cada elemento j, se obtiene el número esperado de fallas y reparaciones durante el periodo entre acciones de reemplazos preventivos $f_j(t_j)$. El intervalo de reemplazos puede ser definido por el número de acciones o reemplazos preventivos x_j durante el tiempo de vida del sistema $T : t_j = T/(x_j + 1)$. Entonces el número esperado de fallas del elemento j durante la vida del sistema es $(x_j + 1) f_j [T/(x_j + 1)]$.

Bajo estas suposiciones, el tiempo esperado del j-ésimo elemento en el que está no disponible puede ser estimado como:

$$(x_j + 1) f_j [T/(x_j + 1)] \tau_{cj} + x_j \tau_{pj}$$

donde τ_{cj} y τ_{pj} son los tiempos de PR y MR respectivamente. Ahora, podemos definir la probabilidad de que cada elemento se encuentre disponible como:

$$p_{j1} = \frac{T - (x_j + 1) f_j [T/(x_j + 1)] \tau_{cj} - x_j \tau_{pj}}{T},$$

así el tiempo total del mantenimiento durante la vida del sistema τ_{tot} es:

$$\tau_{tot} = \sum_{j=1}^{n} \{(x_j + 1) f_j [T/(x_j + 1)] \tau_{cj} + x_j \tau_{pj}\},$$

y el costo esperado del mantenimiento (C_m) durante la vida del sistema es:

$$C_m = \sum_{j=1}^{n} \{(x_j + 1) f_j [T/(x_j + 1)] c_{cj} + x_j c_{pj}\},$$

donde c_{cj} y c_{pj} son los costos de mantenimiento correctivo y preventivo respectivamente.

Teniendo la distribución del tiempo de vida de cada elemento del sistema, es decir, teniendo $g_j = \{0, g_{j1}\}$, $p_j = \{(1 - p_{j1}), p_{j1}\}$; $j = 1, ..., n$, puede obtenerse la distribución del sistema o la función de estructura del mismo. Esto mediante métodos de confiabilidad de diagramas de bloques, al igual que se puede obtener la disponibilidad $A(w, q)$ o la confiabilidad del sistema así como la deficiencia del mismo $D(w, q)$. El costo total de no poder cumplir con toda la demanda durante el tiempo T puede ser estimado como:

$$C_{ud} = T \cdot c_u \cdot D(w, q),$$

donde c_u es el costo específico de no cumplir con la demanda.

Si definimos la política de reemplazo como el vector $x = \{x_j\}, 1 \leq j \leq n$ se pueden tener dos formulaciones para el problema de optimización de mantenimiento.

Formulación 1:

Encontrar el costo mínimo de mantenimiento de cierta política de reemplazos x^* que provea la disponibilidad requerida del sistema multi-estado mientras el tiempo total de mantenimiento no exceda alguna especificación:

$$\begin{aligned} C_m(x^*) &\longrightarrow \text{mín} \\ sujeto\ a\ A(x^*, w, q) &\geq A^*,\ \tau_{tot}(x^*) \leq \tau'. \end{aligned}$$

Formulación 2:

Encontrar la política x^* de reemplazos que minimice el costo total de mantenimiento y de una demanda no satisfecha, mientras que el tiempo total de mantenimiento no exceda cierto límite:

$$\begin{aligned} C_m(x^*) + C_{ud}(x^*, w, q) &\longrightarrow \text{mín} \\ sujeto\ a\ \tau_{tot}(x^*) &\leq \tau'. \end{aligned}$$

En el caso general, se puede usar la siguiente formulación:

$$\begin{aligned} C_m(x^*) + C_{ud}(x^*, w, q) &\longrightarrow \text{mín} \\ sujeto\ a\ A(x^*, w, q) &\geq A^*,\ \tau_{tot}(x^*) \leq \tau', \end{aligned}$$

esta última puede ser reducida a la Formulación 1 definiendo $c_u = 0$ y $A' = 0$.

Dado que se ha definido el problema a optimizar, se procederá a describir el procedimiento que ACO utilizará para obtener la solución requerida.

ACO y metodología a implementar.

Diferentes elementos pueden tener diferentes números de acciones PR durante la vida del sistema. Las posibles alternativas de mantenimiento (número de acciones PR) para cada elemento j pueden ser ordenadas en un vector $Y_j = \{y_{j1}, ..., y_{jK}\}$, donde y_{ji} es el número de acciones de mantenimiento preventivo correspondientes a la alternativa i para el elemento j. El mismo número de K posibles alternativas (longitud del vector Y_j) pueden ser definidas para cada elemento. Si, en la práctica el número de alternativas difiere de elemento a elemento, algunos elementos que posean vectores de longitud corta pueden ser duplicados para proveer vectores de igual longitud.

Cada solución está representada mediante una cadena $a = \{a_1, ..., a_n\}$, donde $1 \leq a_j \leq K$ representa el número de mantenimientos aplicados al elemento j.Luego el

vector x para la solución dada representada por la cadena a es $x = \{y_{1a_1}, ..., y_{na_n}\}$, Por ejemplo para un problema con $n = 5, K = 4$, $Y_1 = Y_2 = Y_3 = \{2, 3, 4, 5\}$ y $Y_4 = Y_5 = \{20, 45, 100, 100\}$ la cadena $a = \{1, 4, 4, 3, 2\}$ representa una solución con $x = \{2, 5, 5, 100, 45\}$. Cualquier cadena entera arbitraria con elementos que pertenecen al intervalo $(1, K)$ representa una solución factible.

Ahora, veamos cómo construiremos dichas cadenas o soluciones. Suponiendo n elementos en el sistema, poseemos los costos, tiempos, tasas de desempeño, función de renovación y diferentes alternativas de mantenimiento por cada elemento del sistema. También conocemos el tiempo de vida del mismo y la distribución de la demanda. Se sabe que para cada elemento existe una cantidad K de mantenimientos distintos. Fijaremos las restrucciones a utilizar de igual modo:

1. En primer lugar incializamos los siguientes parámetros: ρ (tasa de evaporación de feromonas), α (importancia relativa asignada la cantidad de feromonas), β (importancia relativa asignada a la información heurística que se posee), $ants$ (número de hormigas a utilizar), A' el límite mínimo de confiabilidad requerida y τ' el tiempo máximo que debe tardarse la secuencia de mantenimientos.

2. Inicializaremos las siguientes matrices:

$$Matriz\ de\ Información : \eta = \begin{bmatrix} \eta_{11} & \eta_{12} & \cdots & \eta_{1K} \\ \eta_{21} & \eta_{22} & \cdots & \eta_{2K} \\ \vdots & \vdots & \cdots & \vdots \\ \eta_{n1} & \eta_{n2} & \cdots & \eta_{nK} \end{bmatrix};$$

donde η_{ij} representa la información heurística que se posee respecto al mantenimiento $j - ésimo$ asignado al elemento i. Dado que la información que se posee es básicamente el costo por cada mantenimiento, $\eta_{ij} = \frac{1}{1+c_{ij}}$; c_{ij} representa el costo de asignar el mantenimiento j al elemento i.

$$Matriz\ de\ feromonas : \tau = \begin{bmatrix} \tau_{11} & \tau_{12} & \cdots & \tau_{1K} \\ \tau_{21} & \tau_{22} & \cdots & \tau_{2K} \\ \vdots & \vdots & \cdots & \vdots \\ \tau_{n1} & \tau_{n2} & \cdots & \tau_{nK} \end{bmatrix};$$

τ_{ij} representa la cantidad de feromonas depositada en la asignación del mantenimiento j al elemento i. Esta matríz de inicializará con: $\tau_{ij} = \frac{1}{K-1}$, es decir, asignamos la misma proporción a todas las asignaciones posibles. Posteriormente

por cada iteración definida por el usuario, esta matriz de actualizará.

$$Matriz\ de\ probabilidades: P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1K} \\ p_{21} & p_{22} & \cdots & p_{2K} \\ \vdots & \vdots & \ldots & \vdots \\ p_{n1} & p_{n2} & \cdots & p_{nK} \end{bmatrix};$$

p_{ij} representa la probabilidad de asignar el mantenimiento j al elemento i. Siempre se calcularán dichas probabilidades como: $p_{ij} = \frac{\tau_{ij}^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{m \in AC_i} \tau_{im}^{\alpha}[\eta_{im}]^{\beta}}$, donde AC_i es el conjunto de mantenimientos disponibles para el elemento i. De igual modo se actualizará conforme avance el número de iteraciones.

3. Una vez inicializadas las matrices y parámetros necesarios, por cada iteración definidas por el usuario tenemos:

$for\,(i\ \ in\ \ 1:iter)\,\{$

$ACO <- creación\ de\ solución\ por\ hormiga$

$Cost\_Time <- cálculo\ de\ C_m\ y\ \tau_{tot}\ por\ hormiga$

$Best\_Sol <- selección\ de\ la\ mejor\ solución\ de\ las\ ants\ soluciones$

$Tabú <- actualizar\ la\ lista\ tabú\ con\ la\ mejor\ sol.$

$Best\_G <- selecciona\ la\ mejor\ sol.\ global$

$Pherom <- actualización\ de\ la\ matriz\ de\ feromonas$

$P_{transition} <- actualización\ de\ la\ matriz\ de\ probabilidades$

$\}$

Describiremos a continuación el pseudo-código anterior indicando cómo realizar cada uno de los procesos.

ACO: En esta parte se realiza la creación de la secuencia por hormiga. Para cada hormiga a se crea una secuencia de tamaño n de la siguiente forma: Sea x_a la secuencia correspondiente a la hormiga a, entonces se genera un número aleatorio $q \in (0,1)$ tal que para todo elemento i se elige el mantenimiento j:

$$j = \begin{cases} \arg \max\limits_{m \in AC_i} \left[(\tau_{im})^{\alpha}(\eta_{im})^{\beta}\right]; & si\ q \leq q_0 \\ J; & o.c. \end{cases},$$

y J es elegido según la probabilidad:

$$p_{ij} = \begin{cases} \frac{\left[(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}\right]}{\sum_{m \in AC_i}\left[(\tau_{im})^{\alpha}(\eta_{im})^{\beta}\right]}; & si\ j \in AC_i \\ 0; & o.c. \end{cases}.$$

Este procedimiento es realizado n veces hasta completar x_a Se verifica que la solución construida cumpla con las restricciones solicitadas, se calcula $\tau_{tot}(x_a)$, cada $p_{j1}(x_a)$ por elemento, el vector de probabilidades $\underline{p}$ correspondiente a cada combinación de los estados de los elementos mediante el método de diagrama de bloques, calculando mediante composiciones la función de estructura del sistema. Posteriormente se calcula $A(x_a, w)$ con el método de la función generadora universal dada la distribución de la demanda. De esta forma se asegura que cada una de las soluciones generadas satisfaga las restricciones propuestas.

Cost_ Time: Una vez generada una solución por hormiga, se calcula $C_m(x_a)$ y $\tau_{tot}(x_a)$.

Best_ Sol: Se selecciona la mejor solución de las generadas eligiendo aquella que posea el costo mínimo x^*.

Tabú: Se guarda la mejor solución encontrada x^* de todas las iteraciones.

Best_ GS: Se selecciona la mejor solución global X de las encontradas en cada iteración y se guarda.

Pherom: Una vez terminada la selección de la mejor solución, se actualiza la matriz τ de la siguiente forma:

$$Evaporación : \tau_{ij}^{new} = \tau_{ij}^{old}(1-\rho),$$

este proceso se realiza a todos los elementos de la matriz, cada τ_{ij} sufre del proceso de evaporación. Posteriormente se realiza el depósito de feromonas sólamente a aquellos nodos que pertenezcan a la mejor solución encontrada:

$$Depósito : \tau_{ij}^{new} = \tau_{ij}^{old}(1-\rho) + \rho\frac{1}{C_m(x^*)}.$$

$P_{transition}$: Una vez actualizada la matriz de feromonas, se procede a re-calcular las probabilidades de transición correspondientes a la matriz P del mismo modo como fueron calculadas inicialmente.

4. Finalmente, después de todas las iteraciones definidas, se reporta la mejor solución global encontrada así como la confiabilidad que se logra alcanzar y el tiempo total de mantenimiento.

Este algoritmo será desarrollado en R [7]. En los anexos se muestra el código correspondiente y en el próximo capítulo los resultados obtenidos al aplicar ACO, realizándose las comparaciones correspondientes a los resultados propuestos por el autor.

3.1.3. Problema 2: Política de mantenimiento imperfecto óptimo

En el problema anterior, básicamente se consideran acciones de mantenimiento preventivo, cómo se trata de reestablecer los elementos del sistema a sus condiciones inciales, es decir, tan buenos como si fueran nuevos. Sin embargo, existen otras acciones (limpieza, ajuste, etc.) que ayudan a mejorar el desempeño de los elementos del sistema pero no pueden devolverlos a su condición inicial. Tales acciones son llamadas mantenimiento imperfecto preventivo; estos son modelados usando el concepto de reducción de edad.

La evolución de la confiabilidad de los elementos del sistema es una función de la edad de los elementos en la vida de operación de un sistema. La edad de dichos elementos está fuertemente afectada por las activades de mantenimiento del sistema. En el caso en el que los elementos se vuelven tan buenos como si fueran nuevos, su edad se reduce a cero. Sin embargo, existen dos tipos de acciones preventivas, de mantenimiento y de inspección. Cuando se cuenta con activades preventivas que sean de inspección no se afecta el estado de los elementos pero asegura que los elementos se encuentren en condiciones operables, es decir, los elementos se mantienen tan malos como incrementa su edad, pues esta no cambia. Todas las acciones que hacen que la edad de los elementos sea diferente de cero se consideran como acciones PM imperfectas. Cuando un elemento del sistema falla, el mantenimiento correctivo en la forma de reparación mínima se lleva a cabo resultando que el elemento correspondiente esté en condiciones operables sin afectar su tasa de falla.

En este problema, se tratará el problema de la determinación de un plan de acciones PM con costo mínimo durante el tiempo de vida de un sistema multi-estado, el cual provee del nivel de confiabilidad o disponibilidad deseado. Este algoritmo provee de la secuencia en la que deben ser llevadas a cabo las acciones PM, a qué elementos y qué tipo de acción PM de realizará de las disponibles.

Elementos del modelo de reducción de edad

De acuerdo al concepto de reducción de edad, las acciones PM reducen la edad efectiva de el elemento que se encuentra inmediatamente antes de las acciones de mantenimiento. Se usa el modelo de retardo de edad proporcional (Martorell et al, 1999) que asume que la edad efectiva τ_j del elemento j que se encuentra bajo las acciones PM en tiempos cronológicos $(t_{j1}, ..., t_{jn})$ es

$$\begin{aligned} \tau_j(t) &= \tau_j^+(t_{ji}) + (t - t_{ji}) \quad \text{para} \quad t_{ji} < t < t_{ji+1} \quad (0 \le i \le n), \\ \tau_j^+(t_{ji}) &= \epsilon_i \tau_j(t_{ji}) = \epsilon_i \left[\tau_j^+(t_{ji-1}) + (t_{ji} - t_{ji-1})\right], \end{aligned}$$

donde $\tau_j(t_{ji})$ es la edad del elemento inmediatamente despues de la i-ésima acción PM, ϵ_i es el coeficiente de la reducción de edad asociado con la i-ésima acción PM el cual se encuentra en el intervalo $[0,1]$ y $\tau_j(0) = \tau_{j0} = 0$ por definición.

Los dos efectos extremos de PM en el estado del elemento corresponden a los casos cuando $\epsilon = 1$ y $\epsilon = 0$. En el primer caso, el modelo se reduce a "tan malo como viejo", el cual asume que PM no afecta la edad del elemento. En el segundo caso el modelo se reduce a "tan bueno como nuevo", el cual significa que la edad del elemento es reestrablecida a cero (reemplazo). Todas las acciones de PM con $0 < \epsilon < 1$ nos dirigen a una mejora parcial en el estado del elemento.

La introducción de la función de riesgo del elemento j del sistema multiestado puede ser expresasa como:

$$h_j^*(t) = h_j(\tau_j(t)) + h_{j0},$$

donde $h_j(t)$ es la función de riesgo del elemento definida para el caso cuando no se encuentra bajo acciones PM y h_{j0} es el término correspondiente a la edad inicial del elemento, la cual puede diferir de cero al inicio de la vida del MSS.

La confiabilidad del elemento j en el intervalo entre las acciones PM i y $i+1$ es

$$r_j(t) = \exp\left(-\int_{\tau_j^+(t)}^{\tau_j(t)} h_j^*(x)\,dx\right) = \exp\left[H_j\left(\tau_j^+(t_{ji})\right) - H_j(\tau_j(t))\right], \quad t_{ji} \le t \le t_{ji+1}$$

donde $H_j(\tau)$ es la función de riesgo acumulada para el elemento j. Se puede observar el como inmediatamente después de la i-ésima acción PM cuando $t = t_{ji}$ la confiabilidad del elemento es $r_j(t_{ji}) = 1$.

Las reparaciones mínimas son llevadas a cabo si los elementos del sistema multiestado fallan entre las acciones PM. El costo asociado a las reparaciones mínimas depende de las tasas de falla de los elementos. De acuerdo con Boland (1982), el costo mínimo esperado por reparación de el elemento j en el intervalo $[0,t]$ es

$$c_{MR_j}(t) = c_{c_j}\int_0^t h_j(x)\,dx,$$

donde c_{c_j} es el costo de una reparación mínima de este elemento. Cuando un elemento se somete a acciones PM en los tiempos $t_{j1}, ..., t_{jk_j}$, el costo total mínimo por reparación es

$$C_{MR_j} = c_{c_j}\sum_{i=0}^{k_j}\int_{\tau_j^+(t_{ji})}^{\tau_j(t_{ji+1})} h_j(x)\,dx = c_{c_j}\sum_{i=0}^{k_j}\left[H(\tau_j(t_{ji+1})) - H\left(\tau_j^+(t_{ji})\right)\right],$$

donde $t_{j0} = 0$ y $t_{jk_j+1} = T$ por definición.

Formulación del Problema

Se tiene un sistema que consta de n elementos, todos poseen dos posibles estados. Cada elemento j es caracterizado por su tasa de desempeño g_{j1}, la función de riesgo $h_j(t)$ y el costo mínimo de reparación c_{c_j}. El sistema multi-estado debe cubrir cierta demanda w. Las acciones PM y MR pueden ser realizadas en cada elemento del sistema. Las acciones PM pueden modificar la confiabilidad de cada elemento, mientras que las reparaciones mínimas no afectan dicha confiabilidad. La efectividad de cada acción PM está definida por el coeficiente de reducción de edad ϵ, dicho coeficiente se encuentra entre 0 (tan bueno como nuevo) a 1 (tan malo como viejo).

El tiempo en el que el elemento no está disponible debido a actividades MR o PM es insignificante si se compara con el tiempo que transcurre entre actividades consecutivas. Una lista de posibles actividades PM está disponible para cada sistema multi-estado dado. En esta lista, cada acción PM x se encuentra asociada con el costo de su implementación $c_p(x)$, el número de elementos afectados $e(x)$ y el coeficiente de reducción de edad $\epsilon(x)$.

El tiempo de vida T del sistema es dividido en V intervalos, cada uno con duración d_v $(1 \leq v \leq V)$. Las acciones PM pueden ser realizadas en final de cada intervalo. Estas son llevadas a cabo si la confiabilidad del sistema $R(t, w)$ es mas baja que el nivel deseado R'. Debe ser mencionado que d_v de diferentes intervalos no debe ser necesariamente igual y pueden ser elegidos por diferentes razones. Un incremento en el número de intervalos V durante el tiempo de vida del sistema, incrementa la precisión de la solución. Por otra parte, el número de intervalos puede ser limitado por razones tecnológicas.

La secuencia de las acciones PM a ser llevadas a cabo para mantener la confiabilidad MSS es definida como un vector X. Dicho vector posee algúnos números que pertenecen a la lista de acciones PM. Cada acción PM $x \in X$ es necesaria para mejorar la confiabilidad del sistema. La acción a llevarse a cabo es definida por el número siguiente de este vector. Notar que la acción PM elegida x_i puede ser insuficiente para incrementar la confiabilidad MSS al nivel deseado. En este caso, la siguiente acción x_{i+1} debería ser llevada a cabo al mismo tiempo, esto continuamente si R' no es alcanzada. Se puede observar que el número total K de acciones PM no puede ser predefinido y depende de la composición del vector X.

Para un vector X dado, el número total k_j y los tiempos cronológicos de las acciones PM son determinados para cada elemento del sistema j $(1 \leq j \leq n)$. Si $k_j = 0$; que corresponde al caso en el que $e(x_i) \neq j$, $\forall x_i \in x$; el costo mínimo de reparación C_{MR_j} está definido por $t_{j0} = 0$ y $t_{jk_j+1} = t_{j1} = T$. El vector X define mutuamente el costo

de las acciones PM como

$$C_{PM}(x) = \sum_{i=1}^{n} c_p(x_i)$$

y el costo de reparación mínima como

$$C_{MR}(x) = \sum_{j=1}^{n} c_{c_j} \sum_{i=0}^{k_j} \left[H(\tau_j(t_{ji+1})) - H\left(\tau_j^+(t_{ji})\right) \right].$$

El problema es formulado entonces: Encontrar la secuencia óptima x^* de acciones PM elegidas de la lista de acciones disponibles que minimicen el costo total del mantenimiento mientras se provee la confiabilidad MSS requerida

$$\begin{aligned} C_{tot}(x^*) &= C_{PM}(x^*) + C_{MR}(x^*) \longrightarrow \text{mín} \\ s.a.\ \ R(x^*, t, w) &\geq R'; \ \ 0 \leq t \leq T. \end{aligned}$$

Ahora se proseguirá a describir la implementación de ACO para resolver dicho problema de optimización.

ACO y metodología a implementar.

En la metodología a implementar, se espera obtener un vector X que represente la solución encontrada. Dicho vector indica qué acciones PM implementar, en qué orden y a qué elemento debe realizarse. Esto claramente implica que la longitud de cada vector X que haya sido encontrado, sea variable, es decir, dado que existen muchas combinaciones de secuencias y órdenes diferentes para las acciones PM disponibles, no siempre se tendrá un número fijo de acciones a implementar. Es por ello que un número K^* de posiciones redundantes deben estar incluidos en cada vector X.Así, después de construir $X = \{a_1, ..., a_{K^*}\}$, solamente los primeros K números definirán el plan de mantenimiento.

Ahora, veamos cómo construiremos dichos vectores de soluciones. Suponiendo N, el número total de posibles acciones PM, se conoce la distribución de los tiempos de vida por cada elemento j, $1 \leq j \leq n$, los parámentros correspondientes a cada pdf, su tasa de desempeño y el costo de las acciones PM y MR. Se posee una lista de todas las posibles acciones PM indicando para cada uno de los mantenimientos el factor de reducción de edad y el elemento al cual afecta. Para este caso se considera una demanda constante w. Dado el tiempo de vida del sistema T,se particiona el intervalo $[0, T]$ de tal modo que se obtengan subintervalos para analizar el comportamiento y desempeño del sistema. Sea R' el nivel de confiabilidad requerido que siempre debe mantener el sistema.

Antes de implementar ACO, se debe analizar cuál es el primer tiempo en el cual se necesita llevar a cabo la primer acción de mantenimiento. Para esto realizamos lo siguiente:

a) Se calcula la función de estructura ϕ correspondiente al sistema entero, considerando la función de estructura de cada subsistema ϕ_s y las tasas de desempeño g_{j1} por cada elemento.

b) Dada la partición del intervalo $[0,T]$, para cada $t_k \in [0,T]$, se calcula la probabilidad de que cada elemento j no falle sino hasta después de t_k, $P[T_j > t_k] = C_j(t_k)$, así, consideraremos $p_{j1} = C_j(t_k)$ correspondiente a g_{j1}.

c) Usando ϕ, calculamos todos los posibles estados y probabilidades del sistema así como las tasas de desempeño.

d) Calculamos $R(t_k, w)$ para cada $t_k \in [0,T]$ y w dada. Así, para aquel t_k tal que $R(t_k, w) < R'$, es el primer tiempo de mantemiento buscado.

Ya encontrado el tiempo correspondiente al primer mantenimiento, se procede a la implementación de ACO de la siguiente forma:

1. En primer lugar incializamos los siguientes parámetros: ρ, α, β, $ants$.

2. De forma análoga que el problema 1, inicializamos las siguientes matrices:

$$Matriz\ de\ Información : \eta = \begin{bmatrix} \eta_{11} & \eta_{12} & \cdots & \eta_{1N} \\ \eta_{21} & \eta_{22} & \cdots & \eta_{2N} \\ \vdots & \vdots & \ldots & \vdots \\ \eta_{N1} & \eta_{N2} & \cdots & \eta_{NN} \end{bmatrix};$$

donde η_{ij} representa la información heurística de secuenciar después del mantenimiento i el j. Dado que la información que se posee es básicamente el costo por cada mantenimiento, $\eta_{ij} = 1/(c_j + 1)$ es el costo del mantenimiento j, el cual tomamos como el que representa la asignación del j después del mantenimiento i.

$$Matriz\ de\ feromonas : \tau = \begin{bmatrix} \tau_{11} & \tau_{12} & \cdots & \tau_{1N} \\ \tau_{21} & \tau_{22} & \cdots & \tau_{2N} \\ \vdots & \vdots & \ldots & \vdots \\ \tau_{N1} & \tau_{N2} & \cdots & \tau_{NN} \end{bmatrix};$$

τ_{ij} representa la cantidad de feromonas depositada en la asignación del mantenimiento j después del i. Esta matríz de inicializará con: $\tau_{ij} = 1/(N-1)$, es

decir, asignamos la misma proporción a todas las asignaciones posibles. Posteriormente por cada iteración definida por el usuario, esta matriz de actualizará.

$$Matriz\ de\ probabilidades: P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1N} \\ p_{21} & p_{22} & \cdots & p_{2N} \\ \vdots & \vdots & ... & \vdots \\ p_{N1} & p_{N2} & \cdots & p_{NN} \end{bmatrix};$$

p_{ij} representa la probabilidad de asignar el mantenimiento j después del i. Siempre se calcularán dichas probabilidades como: $p_{ij} = \tau_{ij}^{\alpha} \left[\eta_{ij}\right]^{\beta} / \left(\sum_{m \in AC_i} \tau_{im}^{\alpha} \left[\eta_{im}\right]^{\beta}\right)$, donde AC_i es el conjunto de mantenimientos disponibles a asignar después del i. De igual modo se actualizará conforme avance el número de iteraciones.

3. Una vez inicializadas las matrices y parámetros necesarios, por cada iteración tenemos:

$for\,(i\ \ in\ \ 1:iter)\,\{$

$ACO <- creación\ de\ solución\ por\ hormiga$

$Cost\_Time <- cálculo\ de\ C_m\ y\ \tau_{tot}\ por\ hormiga$

$Best\_Sol <- selección\ de\ la\ mejor\ solución\ de\ las\ ants\ soluciones$

$Tabú <- actualizar\ la\ lista\ tabú\ con\ la\ mejor\ sol.$

$Best\_G <- selecciona\ la\ mejor\ sol.\ global$

$Pherom <- actualización\ de\ la\ matriz\ de\ feromonas$

$P_{transition} <- actualización\ de\ la\ matriz\ de\ probabilidades$

$\}$

Describiremos a continuación el pseudo-código anterior indicando cómo realizar cada uno de los procesos.

ACO: En esta parte se realiza la creación de la secuencia por hormiga. Para cada hormiga a se crea una secuencia de tamaño K^* de la siguiente forma: Sea x_a la secuencia de la hormiga a, entonces para elegir qué mantenimiento se realizará se genera un número aleatorio $q \in (0,1)$ tal que se elige el mantenimiento j:

$$j = \begin{cases} \arg \max\limits_{m \in AC_i} \left[(\tau_{im})^{\alpha} (\eta_{im})^{\beta}\right]; & si\ q \leq q_0 \\ J; & o.c. \end{cases},$$

y J es elegido según la probabilidad:

$$p_{ij} = \begin{cases} \frac{\left[(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}\right]}{\sum_{m \in AC_i}\left[(\tau_{im})^{\alpha}(\eta_{im})^{\beta}\right]}; & si\ j \in AC_i \\ 0; & o.c. \end{cases}.$$

Después de haber seleccionado un mantenimiento, se calcula la reducción de edad que sufriría el elemento correspondiente y se vuelve a calcular la confiabilidad $R(t, w)$ del sistema mediante el método de diagrama de bloques encontrando las funciones estructura de cada subsistema y la del sistema entero. Posteriormente se verifica si $R(t, w) > R'$. De ser así, se calcula nuevamente el tiempo en el que es necesario el siguiente mantenimiento de forma análoga al cálculo del primer tiempo de mantenimiento, de lo contrario, se repite el proceso hasta encontrar un mantenimiento k tal que alcance el nivel de confiabilidad requerido o lo supere. Este procedimiento es realizado K^* veces hasta completar x_a.

Cost_ Time: Una vez generada una solución por hormiga, se calcula $C_{tot}(x_a)$.

Best_ Sol: Se selecciona la mejor solución de las generadas eligiendo aquella que posea el costo mínimo x^*.

Tabú: Se guarda la mejor solución encontrada x^* de todas las iteraciones.

Best_ GS: Se selecciona la mejor solución global X de las encontradas en cada iteración y se guarda.

Pherom: Una vez terminada la selección de la mejor solución, se actualiza la matriz τ como en el Problema 1 se definió:

$$Evaporación : \tau_{ij}^{new} = \tau_{ij}^{old}(1 - \rho),$$

realizando este proceso en todos los elementos de la matriz. Posteriormente se realiza el depósito a aquellos nodos que pertenezcan a la mejor solución encontrada:

$$Depósito : \tau_{ij}^{new} = \tau_{ij}^{old}(1 - \rho) + \rho \frac{1}{C_m(x^*)}.$$

$P_{transition}$: Una vez actualizada la matriz de feromonas, se procede a re-calcular las probabilidades de transición correspondientes a la matriz P del mismo modo como fueron calculadas inicialmente.

4. Finalmente, después de todas las iteraciones definidas, se reporta la mejor solución global encontrada así como la confiabilidad que se logra alcanzar y el tiempo total de mantenimiento.

Este algoritmo será desarrollado en R [7]. En el próximo capítulo se incluyen las aplicaciones numéricas de estos problemas y los resultados obtenidos al implementar ACO, realizándose las comparaciones correspondientes a los resultados propuestos por el autor.

Capítulo 4

Implementación y Resultados

Los resultados presentados en este capítulo corresponden a los ejemplos 6.9 y 6.11 de Lisnianski y Levitin [1] como se había mencionado. En ambos casos se realizó la aplicación de ACO que fue detallada en el capítulo anterior. Además se utilizaron los paquetes "snowz "snowfallçreados por Jochen Knaus et al. para R [7]. Estas herramientas sirven para poder ejecutar diversos procesos en paralelo y obtener con una mayor velocidad los resultados finales.

4.1. Problema 1. Mantenimientos perfectos

En esta parte se muestra la aplicación de la metodología descrita para el problema 6.9 en [1], así como una comparación entre los resultados obtenidos en la literatura y los propios. A continuación se tienen detalles del problema.

Se tiene un sistema desalinizador de agua que se encuentra conectado en serie-paralelo. Dicho sistema consiste de 4 componentes, contiene 14 elementos de 8 diferentes tipos. Cada elemento está marcado de acuerdo al tipo que le corresponde.

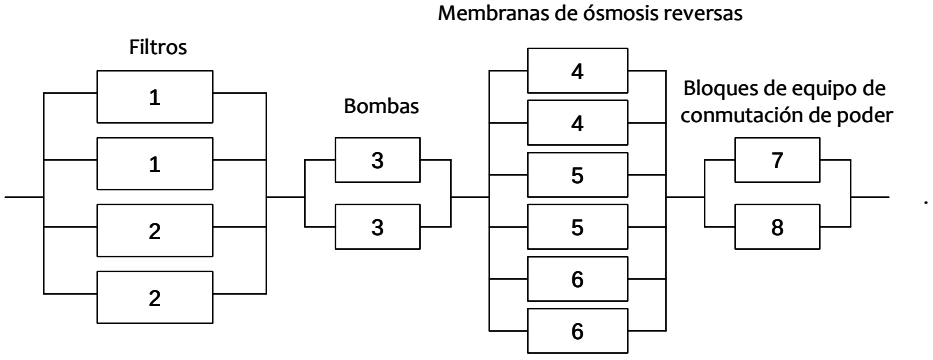

Figura 10 Subsistema de refinaría de petroleo

Se tiene información acerca de la función de renovación correspondiente a cada uno de

los diferentes tipos de elementos. El tiempo es medido en meses. La capacidad nominal por elemento es medida como un porcentaje de la demanda máxima del sistema. Todos los tiempos de reemplazo en el sistema son considerados iguales a 0.5 horas o 0.0007 meses.

					función de renovación $f(t)$					
elemento	g	c_p	c_c	τ_c	$t:24$	12	8	6	4,8	4
					$x:\ 5$	10	15	20	25	30
1	0,40	3,01	0,019	0,002	25	10,0	5,0	2,0	1,00	0,50
2	0,30	2,21	0,049	0,004	26	9,00	2,0	0,6	0,20	0,05
3	0,60	2,85	0,023	0,008	20	4,00	1,0	0,3	0,08	0,01
4	0,15	2,08	0,017	0,005	36	14,0	9,0	6,0	4,00	3,00
5	0,15	1,91	0,029	0,003	55	15,0	7,0	4,0	0,32	0,30
6	0,25	0,95	0,031	0,009	31	9,50	5,6	4,0	2,70	2,00
7	1,00	5,27	0,050	0,002	13	3,20	1,4	0,8	0,50	0,10
8	0,70	4,41	0,072	0,005	5	2,00	1,0	0,4	0,10	0,01

Tabla 4 Características de costo, desempeño y tiempo para cada elemento.

La demanda también juega un papel en este problema y posee su propia distribución. Considerando que el tiempo total de vida del sistema es $T = 120$ meses. El costo de no cumplir el 1 % de la demanda al mes es de $c_u = 10$ unidades.

	Distribución de la demanda			
w	1	0,80	0,50	0,20
q	0,60	0,25	0,05	0,10

Tabla 5 Distribución de la demanda.

El número posible de reemplazos para este caso varía de 5 a 30 con incrementos de 5 unidades. El problema es elegir una política de mantenimientos que indique el número de reemplazos para cada elemento de manera cíclica hasta el término de la vida del sistema como fue indicado. La solución propuesta debe ser tal que cumpla tanto restricciones que tienen que ver con el tiempo de mantenimientos como con el nivel de confiabilidad deseada, buscando siempre reducir los costos de dicha solución propuesta.

Se encontrarán soluciones para las tres formulaciones o escenarios descritos anteriormente en la metodología. Recordemos que cada uno posee una forma distinta de calcular el costo de los mantenimientos:

- Escenario 1: Encontrar el costo mínimo de mantenimiento dada una política x^* que sea capaz de mantener el nivel de confiabilidad deseado mientras el tiempo

de mantenimiento no excede el tiempo mínimo.

$$\begin{aligned} C_m\left(x^*\right) &\longrightarrow \text{mín} \\ s.a. \quad A\left(w, x^*, q\right) &\geq A^{'}, \quad \tau_{tot}\left(x^*\right) \leq \tau^{'}. \end{aligned}$$

- Escenario 2: Encontrar la política de reemplazos óptima x^* tal que minimice el costo de mantenimiento total y el costo de la demanda no es satisfecha, mientras que el tiempo de mantenimiento no excede alguna especificación preliminar.

$$\begin{aligned} C_m\left(x^*\right) + C_{ud}\left(w, x^*, q\right) &\longrightarrow \text{mín} \\ s.a. \quad \tau_{tot}\left(x^*\right) &\leq \tau^{'}. \end{aligned}$$

- Escenario 3: El caso general. Se combinan los escenarios anteriores generando:

$$\begin{aligned} C_m\left(x^*\right) + C_{ud}\left(w, x^*, q\right) &\longrightarrow \text{mín} \\ s.a. \quad A\left(w, x^*, q\right) &\geq A^{'}, \quad \tau_{tot}\left(x^*\right) \leq \tau^{'}. \end{aligned}$$

En la metodología descrita, por cada solución encontrada se calcula y encuentra la función de estructura del sistema entero así como de los subsistemas que por la naturaleza se forman. Este cálculo es necesario, al encontrar un mantenimiento que aplicar por elemento, es posible.calcular la probabilidad correspondiente a g_{j1}, de esta forma encontrar las probabilidades de cada uno de los estados del sistema entero.

En la imagen del sistema, puede apreciarse que existen 4 subsistemas cuyos elementos se encuentran conectados en paralelo. Para cada subsistema se tiene la función de estructura:

1. Subsistema 1: Elementos $\{e_1, e_2, e_3, e_4\}$, los dos primeros del tipo 1 y los segundos del tipo 2. Como cada elemento posee dos posibles estados, el total de estados de este subsistema es $K_1 = 2^4 = 16$. Se tienen 16 diferentes combinaciones de los estados de los 4 elementos correspondientes de acuerdo a sus tasas de desempeño g_{j0} y g_{j1} para $j = 1, 2, 3, 4$. Para obtener la función de estructura asociada, se muestra a continuación un ejemplo del cómo dichas combinaciones generan los estados correspondientes al subsistema 1;

$i = 0, 1$, considerando $G_{S_1} = \phi(g_1, g_2, g_3, g_4)$:

Desempeño por elemento				*Desempeño del Subsistema 1*
g_{1i}	g_{2i}	g_{3i}	g_{4i}	$G_{S_1} = \sum_{j=1}^{4} g_{ji}$
0	0	0	0	0
0,4	0	0	0	0,4
0,4	0,4	0	0	0,8
0,4	0	0,3	0	0,7
0	0,4	0	0	0,4
0,4	0,4	0,3	0	1,1
⋮	⋮	⋮	⋮	⋮
0,4	0,4	0,3	0,3	1,4

Tabla 6 Desempeño del subsistema 1 y sus estados correspondientes.

2. Subsistema 2: Elementos $\{e_5, e_6\}$, ambos del tipo 3. Este subsistema posee $K_2 = 2^2 = 4$ estados, $i = 0, 1$, considerando $G_{S_2} = \phi(g_5, g_6)$:

Desempeño por elemento		*Desempeño del Subsistema 2*
g_{5i}	g_{6i}	$G_{S_2} = \sum_{j=5}^{6} g_{ji}$
0	0	0
0,6	0	0,6
0	0,6	0,6
0,6	0,6	1,2

Tabla 7 Desempeño del subsistema 2 y sus estados correspondientes.

3. Subsistema 3: Elementos $\{e_7, e_8, e_9, e_{10}, e_{11}, e_{12}\}$, los elementos 7 y 8 del tipo 4, 9 y 10 del tipo 5 y los elementos 11 y 12 del tipo 6. Para este subsistema se tienen $K_3 = 2^6 = 64$ posibles estados correspondientes a las combinaciones de las tasas de desempeño g_{j0} y g_{j1} para $j = 7, 8, ..., 12$, $i = 0, 1$ y $G_{S_3} =$

$\phi\left(g_7, g_8, g_9, g_{10}, g_{11}, g_{12}\right)$.

Desempeño por elemento						*Desempeño del Subsistema 3*
g_{7i}	g_{8i}	g_{9i}	g_{10i}	g_{11i}	g_{12i}	$G_{S_3} = \sum_{j=7}^{12} g_{ji}$
0,15	0,15	0	0,15	0,25	0,25	0,95
0,15	0,15	0	0	0,25	0,25	0,8
0,15	0,15	0,15	0	0	0	0,45
0,15	0	0	0	0	0	0,15
0,15	0	0	0	0	0,25	0,40
⋮	⋮	⋮	⋮	⋮	⋮	⋮
0	0	0	0,15	0	0	0,15

Tabla 8 Desempeño del subsistema 3 y sus estados correspondientes.

4. Subsistema 4: Elementos $\{e_{13}, e_{14}\}$, cada uno del tipo 7 y 8 respectivamente. Para este subsistema se tienen $K_4 = 2^2 = 4$ posibles estados correspondientes a las combinaciones de las tasas de desempeño g_{j0} y g_{j1} para $j = 13, 14$, $i = 0, 1$ considerando $G_{S_3} = \phi\left(g_{13}, g_{14}\right)$:

Desempeño por elemento		*Desempeño del Subsistema 4*
g_{13i}	g_{14i}	$G_{S_3} = \sum_{j=13}^{14} g_{ji}$
0	0	0
1	0	1
0	0,7	0,7
1	0,7	1,7

Tabla 9 Desempeño del subsistema 4 y sus correspondientes estados.

Así, para obtener el desempeño del sistema utilizando la función de estructura, dado que ya se tiene el desmpeño de cada subsistema, tenemos entonces $K_S = 16 \times 4 \times 64 \times 4 = 16384$ estados del sistema que se calculan como $G_S = \phi\left(G_{S_1}, G_{S_2}, G_{S_3}, G_{S_4}\right)$:

Desempeño por subsistema				*Desempeño del sistema*
G_{S_1}	G_{S_2}	G_{S_3}	G_{S_4}	$G_S = \text{mín}\left\{G_{S_1}, G_{S_2}, G_{S_3}, G_{S_4}\right\}$
0	0	0	0	0
0,4	0,6	0,8	1	0,4
0,8	1,2	0,15	0,7	0,15
⋮	⋮	⋮	⋮	⋮
1,4	0,6	0,15	1,7	0,15

Tabla 10 Desempeño del sistema y sus correspondientes estados.

En términos generales la función de estructura del sistema se obtiene

$$\begin{aligned} G_S &= \phi\left(G_{S_1}, G_{S_2}, G_{S_3}, G_{S_4}\right) \\ &= \phi\left(\phi\left(g_1, g_2, g_3, g_4\right), \phi\left(g_5, g_6\right), \phi\left(g_7, g_8, g_9, g_{10}, g_{11}, g_{12}\right), \phi\left(g_{13}, g_{14}\right)\right) \\ &= \phi\left(\sum_{j=1}^{4} g_{ji}, \sum_{j=5}^{6} g_{ji}, \sum_{j=7}^{12} g_{ji}, \sum_{j=13}^{14} g_{ji}\right) \\ &= \text{mín}\left\{\sum_{j=1}^{4} g_{ji}, \sum_{j=5}^{6} g_{ji}, \sum_{j=7}^{12} g_{ji}, \sum_{j=13}^{14} g_{ji}\right\};\ i = 0, 1. \end{aligned}$$

Asociado a cada uno de los elementos del sistema, se tienen las funciones generadoras universales que relacionan sus tasas de desmpeño con las probabilidades de los posibles estados. Cada función posee la forma

$$u_j\left(z\right) = \sum_{i=1}^{k_j} p_{ij} z^{g_{ij}},$$

donde k_j son los posibles estados relacionados al elemento j. Dicha función generadora es obtenida una vez que son calculados los correspondientes p_{ij} de cada elemento según el mantenimiento elegido. Después, la combinación de cada opción por elemento, mediante la función de estructura, da lugar a los p_i de cada estado del sistema total.

Es claro que este problema no posee tantas exigencias y requerimientos como el segundo, sin embargo, se decidió diversificar la búsqueda de soluciones en el espacio de las mismas para tratar de mejorar las propuestas por los autores o igualarlas. Es por esta razón por la que la programación en paralelo resultó ser una herramienta útil. Se realizaron diferentes simulaciones variando los parámetros de ACO para cada uno de los escenarios propuestos por el autor en los cuales se puede minimizar el costo bajo diferentes restricciones. Dado que el problema no es el cómo se comporta ACO y las variaciones de sus parámetros, se presentarán sólamente aquellos resultados significativos y bajo qué condiciones fueron encontrados.

Escenario 1

A continuación se presentan las soluciones encontradas utilizando la metodología propuesta para este problema. Se tiene la restricción: $A\left(x^*, w, q\right) \geq 0{,}96$.

	C_{ud}	C_m	$C_{ud} + C_m$	τ_{tot}	A	x
			Solución de Lisnianski y Levitin			
$A' = 0{,}96$	0,0	263,1	263,1	9,2	0,9606	$\{5,5,5,5,5,5,5,5$ $10,10,10,10,5,5\}$
			Mejor solución propia encontrada			
$A' = 0{,}96$	0,0	263,061	263,061	9,168	0,9606386	$\{5,5,5,5,5,5,5,5$ $10,10,10,10,5,5\}$

Puede observarse que se igualó la solución del autor, mostrándose una mayor precisión en cuanto a los números y el cómo ACO, mediante la metodología propuesta, funciona correctamente. Por otra parte, cabe mencionar que se realizaron varias simulaciones eligiendo aquellos parámetros que permitían acercarse más rápido a la solución óptima.

La selección de estos parámetros fue realizada jugando con las combinaciones de estos mismos. Para este caso se utilizaron 10 hormigas, 300 iteraciones con parámetros $\rho = 0{,}2$, $\alpha = 0{,}3$ y $\beta = 0{,}65$.

Cabe mencionar que este problema fue simulado en una máquina con 4 procesadores teniendo un tiempo de ejecución de 7.16 horas. Intuitivamente ρ funciona mucho mejor con valores más chicos a 0.5, pues al tener un comportamiento de tasa de descuento, entre menor sea, esta, mayor será la posibilidad de extender el número de iteraciones necesarias para llegar a converger a alguna solución; de otro modo; al descontar mucho más cantidad de feromonas, se podría tener convergencia rápida a una solución que muy probablemente sea un óptimo global. De este modo, se trató de diversificar el portafolio de soluciones encontradas por iteración aumentando tanto el número de hormigas como disminuyendo ρ.

En tanto a α, empíricamente se verificó que para $\alpha < \beta$, se obtienen mejores resultados. Esto parece lógico pues dado que α representa la importancia que se le da a la información generada por el rastro de feromonas, entre mayor sea α, esta información se reducirá y tendrá menor peso en la elección y actualización de las probabilidades según la metodología propuesta.

Con la restricción $A(x^*, w, q) \geq 0{,}97$ se obtiene:

	C_{ud}	C_m	$C_{ud} + C_m$	τ_{tot}	A	x
			Solución de Lisnianski y Levitin			
$A' = 0{,}97$	0,0	296,6	296,6	7,7	0,9700	$\{5, 5, 5, 5, 10, 10, 5, 5$ $10, 10, 25, 25, 5, 5\}$.
			Mejor solución propia encontrada			
$A' = 0{,}97$	0,0	301,217	301,217	7,65	0,9704	$\{5, 5, 5, 5, 10, 10, 5, 5$ $10, 10, 20, 15, 5, 5\}$

Para esta solución se utilizaron 10 hormigas, $\rho = 0{,}3$, $\alpha = 0{,}2$ y $\beta = 0{,}4$. El tiempo de ejecución fue de 4.5 horas con 200 iteraciones. Esta solución está muy cercana a la encontrada por el autor, sin embargo, dado que se realizaron varias pruebas con combinaciones distintas de los parámetros de ACO, la presentada es la mejor solución propuesta.

Con la restricción $A(x^*, w, q) \geq 0{,}98$ se obtiene:

	C_{ud}	C_m	$C_{ud}+C_m$	τ_{tot}	A	x
			Solución de Lisnianski y Levitin			
$A' = 0{,}98$	0,0	384,4	384,4	5,85	0,9800	$\{5,5,5,5,15,15,10,$ $10,10,25,25,25,25,5\}$.
			Mejor solución propia encontrada			
$A' = 0{,}98$	0,0	437,52	437,52	5,42	0,9805	$\{15,5,10,5,15,20,10,10$ $25,25,10,15,5,5\}$

Para este caso, se tiene una solución cuyo costo no es menor que el propuesto en la solución de los autores, sin embargo, las restricciones de tiempo y de nivel de confiabilidad son mejores a los dados. Para este caso, con 100 iteraciones y 10 hormigas, el tiempo de ejecución fueron 2.25 horas. Es clara la disminución de tiempo, esto se debe a los parámetros utilizados $\rho = 0{,}385$, $\alpha = 0{,}4$ y $\beta = 0{,}099$. Estos parámetros propuestos resultaron de la experiencia y conocimiento empírico desarrollado para la aplicación de ACO en este problema.

Escenario 2

En este caso se minimiza el costo de aplicar cierta política de mantenimientos y el costo generado por no satisfacer la demanda en el momento de realizar los mantenimientos. La única restricción a considerar es el tiempo de ejecución, es decir minimizar C_m.

Las soluciones de los autores y la propia son presentadas a continuación:

		C_{ud}	C_m	$C_{ud}+C_m$	τ_{tot}	A	x
				Solución de Lisnianski y Levitin			
Mínimo	C_m	1029,5	249,1	1278,6	11,61	0,9490	$\{5,5,5,5,5,5,5,$ $5,5,5,5,5,5,5\}$.
				Mejor solución propia encontrada			
Mínimo	C_m	9,8755	301,59	311,46	10,27	0,9549	$\{10,5,5,5,5,5,10,$ $5,25,5,5,5,5,5\}$

La solución que se encontró utilizando ACO es mucho mejor que la propuesta por los autores, tanto en los costos como en el tiempo de ejecución y el nivel de confiabilidad alcanzado. De nuevo, dado el conocimiento empírico, la ejecución de este problema tuvo un tiempo de 1.11 horas. Bastaron 50 iteraciones con 10 hormigas para converger a una buena solución. Esto puede atribuirse a la elección de los parámetros $\rho = 0{,}2$, $\alpha = 0{,}6$ y $\beta = 0{,}1$. Estos parámetros fueron encontrados después de realizar varias pruebas.

Escenario 3

Este caso trata la formulación general, en la cual, se busca minimizar el costo total de una política de mantenimiento, incluyendo tanto C_m como C_{ud}. Se tienen las dos restricciones del escenario 1, la confiabilidad no debe ser menor a un cierto nivel dado y el tiempo de ejecución debe ser manor o igual al propuesto. En este caso se tiene:

	C_{ud}	C_m	$C_{ud}+C_m$	τ_{tot}	A
			Solución de Lisnianski y Levitin		
Mínimo C_m+C_{ud}	192,8	498,1	690,9	4,96	0,9850
s.a. $\tau'=5{,}5,\ A'=0{,}985$		x	$\{5,5,5,5,25,25,10,$	$10,25,25,30,30,10,5\}$	
			Mejor solución propia encontrada		
Mínimo C_m+C_{ud}	2,399	509,25	511,65	4,72	0,9855
s.a. $\tau'=5{,}5,\ A'=0{,}985$		x	$\{5,5,5,5,20,15,25,$	$25,25,25,30,25,5,10\}$	

Este caso puede decirse, resulta ser el más importante y el más completo. Dado que posee ambas restricciones y el costo total a minimizar resulta de la suma de los costos relacionados a los mantenimientos y al no satisfacer cierta demanda, podría decirse que la solución encontrada es mucho más completa en los aspectos requeridos. La solución obtenida es mejor a la propuesta por los autores.

El escenario anterior se obtuvo con 300 iteraciones y 10 hormigas, se utilizaron $\alpha = 0{,}4$, $\rho = 0{,}385$ y $\beta = 0{,}0999$. El tiempo de ejecución fue de 11.24 horas. Nuevamente se aprovechó el conocimiento adquirido en este problema acerca del rango de valores que podrían ser utilizados en los parámentros, esto con la finalidad de realizar una búsqueda inteligente y guiada.

Finalmente, puede observarse que ACO resultó una herramienta apropiada para resolver este problema. La limitante en este caso sería el encontrar parámentros adecuados tales que ayuden a encontrar una buena solución y se logre la convergencia.

4.2. Problema 2. Mantenimientos imperfectos

Tal y como fue descrito en el capítulo anterior, en esta sección se muestra la aplicación al problema 6.11 de Lisnianski y Levitin, de la metodología propuesta, así como una comparación entre los resultados obtenidos por el autor y los propios. Se decribe a continuación detalles del problema.

Se tiene un sistema de refineria de petróleo que posee una estructura serie-paralelo, el cual, consta de cuatro componentes conectados en serie como se muestra a continuación. El sistema pertenece al tipo de sistemas MSS de transmisión y dispersión. Este posee 11 elementos bi-estado con diferentes tasas o niveles de desempeño y funciones

de confiabilidad.

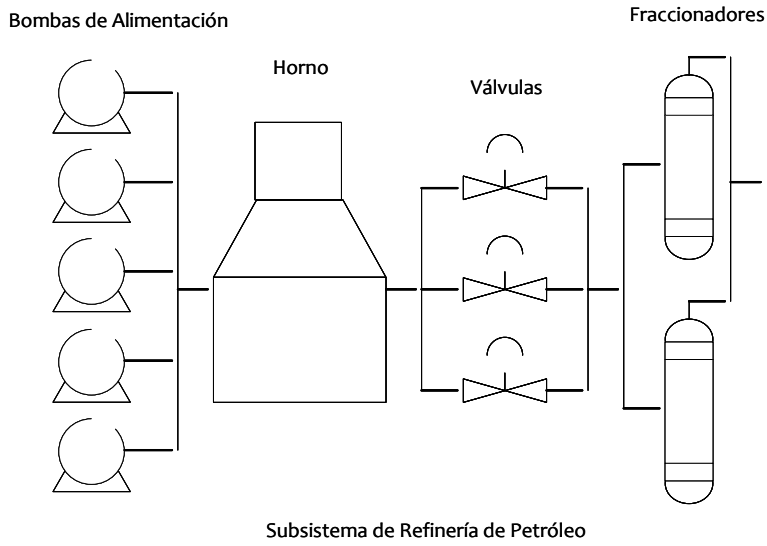

Figura 11 Sistema serie-paralelo en refinería de petroleo.

La confiabilidad de cada elemento j es definida por una función de intensidad Weibull

$$h_j(t) = \lambda^{\gamma}\gamma\left[\tau(t)\right]^{\gamma-1} + h_0,$$

la cual es comunmente utilizada en equipos reparables. La función de riesgo acumulada toma la forma:

$$H(t) = \left[\lambda\tau(t)\right]^{\gamma} + h_0\tau(t).$$

La tasa nominal de desempeño por elemento g_{j1},el parámetro de escala de la función de intensidad λ_j, el parámetro de forma γ_j y la constante de riesgo h_{j0} se muestran en la siguiente tabla para cada elemento j. Esta tabla también contiene el costo de

reparaciones mínimas por cada elemento.

Elemento	g_{j1}	λ_j	γ_j	h_{j0}	Costo MR
1	0,4	0,050	1,8	0,0001	0,9
2	0,4	0,050	1,8	0,0001	0,9
3	0,4	0,050	1,8	0	0,9
4	0,4	0,070	1,2	0,0003	0,8
5	0,6	0,010	1,5	0	0,5
6	1,3	0,010	1,8	0,00007	2,4
7	0,6	0,020	1,8	0	1,3
8	0,5	0,008	2,0	0,0001	0,4
9	0,4	0,020	2,1	0	0,7
10	1,0	0,034	1,6	0	1,2
11	1,0	0,008	1,9	0,0004	1,9

Tabla 11 Parámetros de interés por elemento.

El conjunto de acciones de mantenimientos preventivos (PM) posibles, son definidos a continuación. Cada acción es caracterizada por su costo, el elemento al cual afecta o se es aplicado y el coeficiente de reducción de edad ϵ. Las acitividades PM pueden incluir reemplazos $(\epsilon = 0)$, sobrevivencia $(\epsilon = 1)$ y acciones imperfectas que proveen de un efecto parcial de mejora $(0 < \epsilon < 1)$.

Acción PM	Elemento afectado	Reducción ϵ $(Edad)$	Costo de PM	Acción PM	Elemento afectado	Reducción ϵ $(Edad)$	Costo de PM
1	1	1,00	2,2	16	6	0,00	19,0
2	1	0,56	2,9	17	7	0,75	4,3
3	1	0,00	4,1	18	7	0,00	6,5
4	2	1,00	2,2	19	8	0,80	5,0
5	2	0,56	2,9	20	8	0,00	6,2
6	2	0,00	4,1	21	9	1,00	3,0
7	3	1,00	2,2	22	9	0,65	3,8
8	3	0,56	2,9	23	9	0,00	5,4
9	3	0,00	4,1	24	10	1,00	8,5
10	4	0,76	3,7	25	10	0,70	10,5
11	4	0,00	5,5	26	10	0,00	14,0
12	5	1,00	7,3	27	11	1,00	8,5
13	5	0,60	9,0	28	11	0,56	12,0
14	5	0,00	14,2	29	11	0,00	14,0
15	6	0,56	15,3				

Tabla 12 Listado de acciones PM disponibles y características de cada una.

El tiempo de vida del sistema es de 25 años. Los tiempos para las posibles acciones PM están espaciados en intervalos de $d = 1{,}5$ meses $(0{,}125\ años)$. El problema es generar una secuencia de acciones PM que permita al sistema seguir trabajando con un desempeño tal que el nivel de confiabilidad no sea menor a un límite R' y se satisfaga siempre cierto porcentaje w de la demanda.

Para ver la influencia de los límites asociados a las restricciones de demanda y nivel de confiabilidad $(w \ \ y \ \ R')$, se presentan cuatro escenarios con variedad de dichos niveles. En cada uno de estos es obtenida una secuencia óptima. Cada escenario es una combinación de los valores: $w = 0{,}8, 1$ y $R' = 0{,}9,\ 0{,}95$. Las secuencias obtenidas serán presentadas a continuación precedidas por un análisis del comportamiento del sistema sin considerar ningún tipo de acción de mantenimiento.

Como puede observarse en la gráfica del sistema, este puede ser dividido en 4 subsistemas cuyos elementos están conectados en paralelo. Para facilitar el cálculo de la función de estructura del sistema entero, se trabajó primero con cada subsistema para después conjuntar todo en una sola expresión. Así, para cada uno se tiene:

1. Subsistema 1: Elementos $\{e_1, e_2, e_3, e_4, e_5\}$. Como cada elemento posee dos posibles estados, el total de estados de este subsistema es $K_1 = 2^5 = 32$. Esto quiere decir que se tienen 32 diferentes combinaciones de los estados de los 5 elementos correspondientes de acuerdo a sus tasas de desempeño g_{j0} y g_{j1} para $j = 1, 2, 3, 4, 5$. La siguiente tabla muestra dichas combinaciones y la tasa de desempeño del susbistema asociado a cada uno de sus 32 estados con función de desempeño $G_{S_1} = \phi(g_1, g_2, g_3, g_4, g_5)$ para $i = 0, 1$.

Desempeño por elemento					Desempeño del Subsistema 1
g_{1i}	g_{2i}	g_{3i}	g_{4i}	g_{5i}	$G_{S_1} = \sum_{j=1}^{5} g_{ji}$
0	0	0	0	0	0
0,4	0	0	0	0	0,4
0,4	0,4	0	0	0	0,8
0,4	0	0,4	0	0,4	1,2
0,4	0	0	0	0,6	1
0,4	0,4	0,4	0	0,6	1,8
⋮	⋮	⋮	⋮	⋮	⋮
0,4	0,4	0,4	0,4	0,6	2,2

Tabla 13 Desempeño del subsistema 1 y sus correspondientes estados.

2. Subsistema 2: Elementos $\{e_6\}$. Este subsistema consta de un solo elemento, esto

implica solamente posee dos estados, así $G_{S_2} = \phi(g_6) = g_{6i}$, para $i = 0, 1$.

Desempeño elemento	Desempeño Subsistema 2
g_{6i}	$G_{S_2} = g_{6i}$
0	0
1,3	1,3

Tabla 14 Desempeño del subsistema 1 y estados.

3. Subsistema 3: Elementos $\{e_7, e_8, e_9\}$. Para este subsistema se tienen $K_3 = 2^3 = 8$ posibles estados correspondientes a las combinaciones de las tasas de desempeño g_{j0} y g_{j1} para $j = 7, 8, 9$, donde $G_{S_3} = \phi(g_7, g_8, g_9)$, para $i = 0, 1$.

Desempeño por elemento			Desempeño del Subsistema 3
g_{7i}	g_{8i}	g_{9i}	$G_{S_3} = \sum_{j=7}^{9} g_{ji}$
0	0	0	0
0,6	0	0	0,6
0	0,5	0	0,5
0	0	0,4	0,4
0,6	0,5	0	1,1
0	0,5	0,4	0,9
0,6	0	0,4	1
0,6	0,5	0,4	1,5

Tabla 15 Desempeño del subsistema 3 y correspondientes estados.

4. Subsistema 4: Elementos $\{e_{10}, e_{11}\}$. Para este subsistema se tienen $K_4 = 2^2 = 4$ posibles estados correspondientes a las combinaciones de las tasas de desempeño g_{j0} y g_{j1} para $j = 10, 11$, $i = 0, 1$. La función de estructura del subsistema es $G_{S_3} = \phi(g_{10}, g_{11})$.

Desempeño por elemento		Desempeño del Subsistema 4
g_{10i}	g_{11i}	$G_{S_3} = \sum_{j=10}^{11} g_{ji}$
0	0	0
1	0	1
0	1	1
1	1	2

Tabla 16 Desempeño del subsistema 4 y sus correspondientes estados.

Así, para obtener el desempeño del sistema utilizando la función de estructura, dado que ya se tiene el correspondiente a cada subsistema, tenemos entonces $K_S = 32 \times 2 \times$

$8 \times 4 = 2048$ estados del sistema que se calculan como $G_S = \phi\left(G_{S_1}, G_{S_2}, G_{S_3}, G_{S_4}\right)$.

Desempeño por subsistema				Desempeño del sistema
G_{S_1}	G_{S_2}	G_{S_3}	G_{S_4}	$G_S = \text{mín}\left\{G_{S_1}, G_{S_2}, G_{S_3}, G_{S_4}\right\}$
0	0	0	0	0
0	0	0	1	0
0,4	1,3	0,4	2	0,4
⋮	⋮	⋮	⋮	⋮
2,2	1,3	1,5	2	1,3

Tabla 17 Desempeño del sistema y sus correspondientes estados.

En términos generales la función de estructura del sistema se obtiene:

$$\begin{aligned} G_S &= \phi\left(G_{S_1}, G_{S_2}, G_{S_3}, G_{S_4}\right) \\ &= \phi\left(\phi\left(g_1, g_2, g_3, g_4, g_5\right), \phi\left(g_6\right), \phi\left(g_7, g_8, g_9\right), \phi\left(g_{10}, g_{11}\right)\right) \\ &= \phi\left(\sum_{j=1}^{6} g_{ji}, g_6, \sum_{j=7}^{9} g_{ji}, \sum_{j=10}^{11} g_{ji}\right) \\ &= \text{mín}\left\{\sum_{j=1}^{6} g_{ji}, g_6, \sum_{j=7}^{9} g_{ji}, \sum_{j=10}^{11} g_{ji}\right\};\ i = 0, 1. \end{aligned}$$

Asociado a cada uno de los elementos del sistema, se tienen las funciones generadoras universales (UGF) que relacionan sus tasas de desmpeño con las probabilidades de los posibles estados, cada función posee la forma:

$$u_j\left(z\right) = \sum_{i=1}^{k_j} p_{ij} z^{g_{ij}},$$

donde k_j son los posibles estados relacionados al elemento j. Esta función universal también existe para cada subsistema y para el sistema entero.

Se calculó el desempeño del sistema basado en las distribuciones del tiempo de vida de cada elemento, es decir, utilizando las distribuciones Weibull y las funciones de riesgo acumuladas, sin considerar que debe ser satisfecha ninguna demanda se tiene que la confiabilidad para cada elemento está dada por

$$C\left(t\right) = \exp\left[-H\left(t\right)\right],$$

con las particularidades descritas de dicha función acumulada. La siguiente gráfica

muestra dicho desempeño.

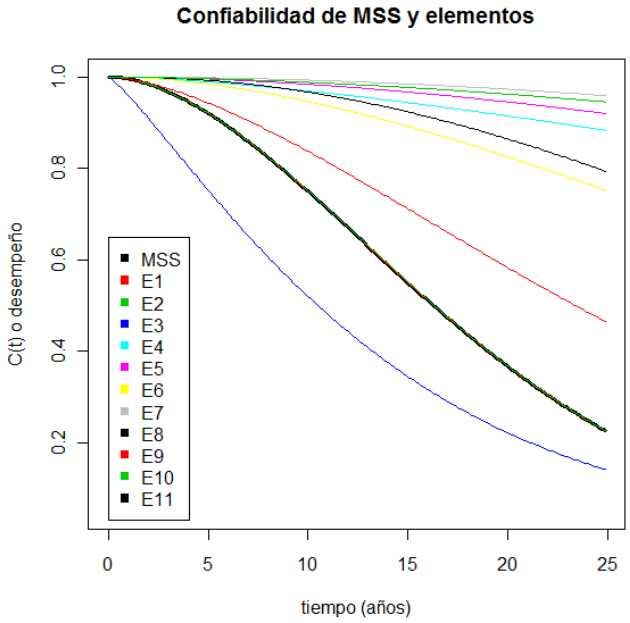

Figura 12 Desempeño de la confiabilidad por elemento y sistema.

Como puede observarse la línea cuyo grosor es mayor a las demás representa el desempeño del sistema en su conjunto, mientras que las otras líneas delgadas y de distintos colores representan el desempeño de cada uno de los elementos que lo componen. Se observa que entre los años 5 y 10 la confiabilidad del sistema decrece rápidamente, mientras que apartir del año 10 y en adelante, dichos niveles caen considerablemente a niveles muy bajos. El desempeño se ve afectado básicamente por el de los elementos 1, 9, 2 y 3. Los demás elementos parecieran mantener un decremento no drástico comparado con el de los otros. Cabe mencionar que el cálculo de dichas confiabilidades no fue hecho utilizando la función generadora universal, pues simplemente se quiere mostrar el cómo baja dicho desempeño.

A continuación se expondrán los 4 escenarios propuestos por Lisnianski y Levitin [1]. Se incluirán gráficas del cómo y en qué momento es necesario implementar acciones PM y qué es lo que se desea evitar. Se compararán las soluciones con las del libro.

4.2.1. Escenario 1

En este caso se consideran unos límites de $w = 0{,}8$ y $R' = 0{,}9$, es decir, se desea que el sistema cumpla con el 80 % de la demanda solicitada con un nivel de confiabilidad no menor a un 90 %. Para el cálculo de dicha confiabilidad considerando el factor

demanda, se utilizó el método para construir la función de estrucutra del sistema y la función generadora universal para incluir dicho factor.

Así, para este caso, el tiempo al cual se debe realizar el primer mantenimiento o acción PM es $t_1 = 14{,}250$. La siguiente gráfica muestra el desempeño del sistema bajo estas consideraciones.

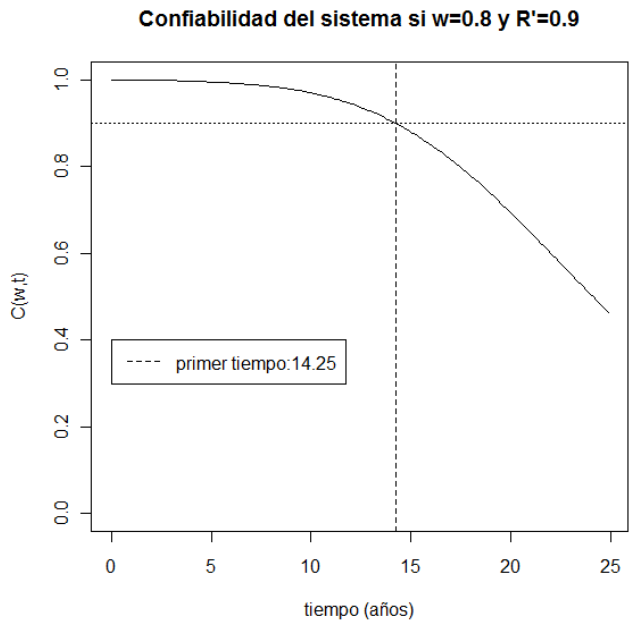

Figura 13 Comportamiento de la confiabilidad del sistema.

Como puede observarse el desempeño del sistema considerando la demanda presenta cambios significativos e indica el momento al cual la confiabilidad del sistema se encuentra por caer por debajo del 90 %.

Nuevamente recordamos el problema a optimizar:

$$\begin{aligned} \text{mín } C_{tot}(x^*) &= \text{mín } \{C_{PM}(x^*) + C_{MR}(x^*)\} \\ s.a.\ \ R(x^*, t, 0{,}8) &\geq 0{,}90; \quad 0 \leq t \leq 25. \end{aligned}$$

Aprovechando la naturaleza estocástica de ACO, se hubiera deseado obtener resultados que provengan de simulaciones con un mayor número de iteraciones, pero aun utilizando programación en paralelo con 4 procesadores, el tiempo de ejecución resultó ser alto como se muestra en la siguiente tabla. Dada esta premisa, el momento en el que se dejó de realizar simulaciones en este escenario fue cuando se encontró una mejor

solución propia la cual se muestra a continuación:

$iteraciones = 100$				
C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	14,250	8	3	0,9495
	15,625	5	2	0,9405
	16,750	7	3	0,9431
	16,750	18	7	0,9431
53,062	17,50	16	6	0,9412
	19,250	8	3	0,9685
	21,000	5	2	0,9628
	22,625	7	3	0,9627
	22,625	6	2	0,9627
$burning\ \ time$: 3,8 hrs				

Tabla 18*a* Solución encontradas en 100 iteraciones

$iteraciones = 500$				
C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	14,250	3	1	0,9495
	17,750	1	1	0,9058
	17,750	8	3	0,9058
	18,500	17	7	0,9205
49,1626	18,875	6	2	0,9206
	20,500	16	6	0,9565
	24,375	1	1	0,9246
	24,375	8	3	0,9246
$burning\ \ time$: 26 hrs				

Tabla 18*b* Solución encontradas en 100 iteraciones

Para ambos casos se presentan las siguientes gráficas del comportamiento de la confiabilidad a través del tiempo de vida del sistema. Se puede observar el cómo aumenta en cada tiempo marcado evitando así que baje del nivel R' dado. Una aclaración importante, en las gráficas de resultados se muestra la variable "tiempo.[en] el eje X, la unidad que se maneja son años. Se verá el mismo formato en todas las gráficas ya que fueron generadas de modo genérico, sin embargo en el pie de cada imagen se dan

detalles sobre cada una.

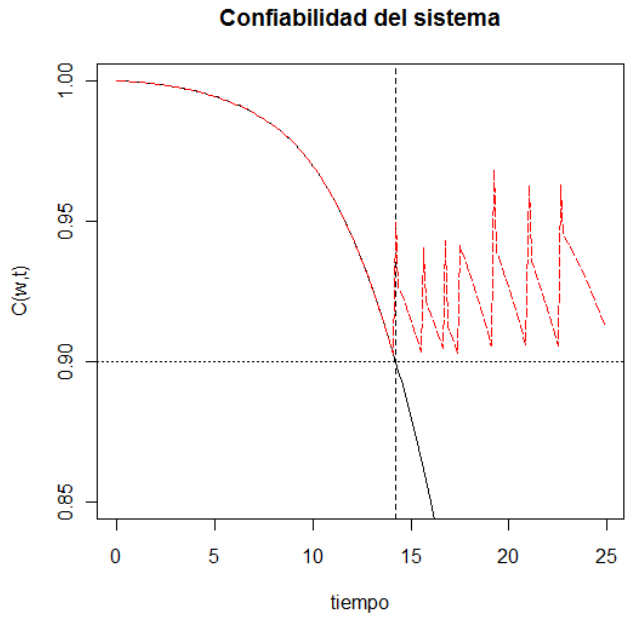

Figura 14 Confiabilidad del sistema dada la primera solución

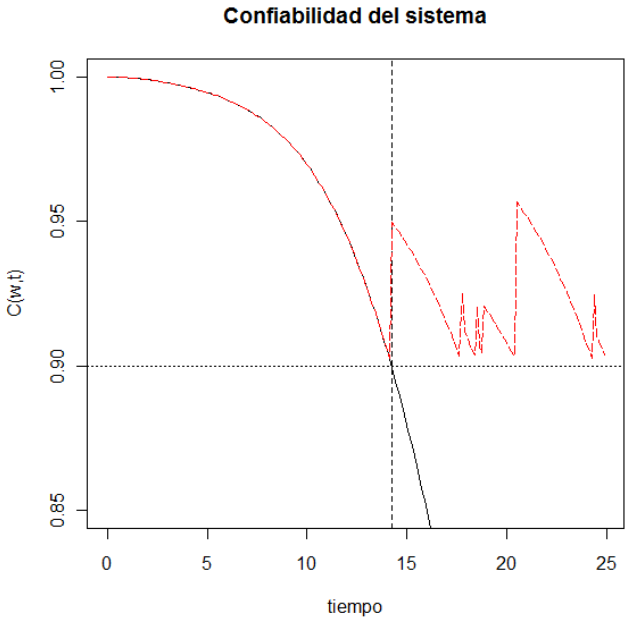

Figura 15 Confiabilidad del sistema con la segunda solución.

La línea punteada corresponde a los incrementos de la confiabilidad después de haber aplicado algún mantenimiento a los elementos dados, nótese que en las soluciones hay dos mantenimientos indicados con el mismo tiempo, esto es, se deberán hacer dichos

mantenimientos en el mismo tiempo.

A continuación se presenta la mejor solución encontrada en [1] para este escenario.

C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	14,250	6	2	0,949
	17,875	8	3	0,923
34,824	19,500	15	6	0,948
	21,750	21	9	0,932
	23,000	2	1	0,947

Tabla 19 Solución presentada por Lisnianski y Levitin.

Comparando esta solución y la propia, claramente se observa que aún no se logra mejorar. Se piensa que la solución propia es la mejor como óptimo local. Cabe aclarar que ACO funciona con tres parámetros de intensidad tal y como se mencionó en el problema anterior: α, β y ρ. Estos miden el grado de importancia que se le dará a la matriz de feronomas, a la matriz de información y a la rapidez con la que se evaporará la cantidad de feromonas de los caminos menos visitados. El espacio para cada parámetro resulta bastante amplio, tanto que para cada terna, resulta complicado y difícil de explorar. Estos tres parámetros reciben valores dados por el usuario.

La terna ideal sería aquella que ayudara a incrementar la velocidad de la convergencia a la solución óptima global; es decir; la terna de valores óptimos resulta también un problema de optimización combinatoria que vive en tres dimensiones. El análisis correspondiente a dichos parámetros no es estudiado en este trabajo, sin embargo, se hicieron diferentes pruebas variando dicha terna y empíricamente se ha tratado de encontrar cierta combinación que generara una mejor búsqueda.

Todos los escenarios en los cuales se utilizaron 4 procesadores tuvieron valores para $\rho = 0{,}5$, $\alpha = 0{,}4$ y $\beta = 0{,}45$. Esto con la finalidad de darle el mismo peso al depósito de feronomas como a la evaporación, un poco más de importancia a la toma de decisiones basada en la matriz de feromonas que a la de información.

Dado que el tiempo de ejecución continua siendo grande, se procedió a realizar la programación en paralelo en una máquina con 16 procesadores (Bitachi). Lo que se gana en este caso además de velocidad en el procesamiento, es la diversificación de búsquedas de soluciones, pues con 4 procesadores siempre se utilizaron 4 u 8 hormigas o agentes de búsqueda. En el caso de Bitachi, el número mínimo de agentes es de 16, utilizando en ocasiones hasta 32 hormigas. Esto ayuda a diversificar los puntos de búsqueda y a encontrar más rápido una buena solución. Cabe aclarar que por comodidad y mayor uso de los recursos computacionales siempre se utilizaron número de agentes múltiplo de los procesadores existentes.

Durante la ejecución del programa en Bitachi, se identificó que el número de iteraciones para converger a una solución se redujeron al mismo tiempo que se variaban los parámetros de ACO. A continuación se presentan las soluciones mas relevantes del proceso en Bitachi:

$Solución\ 1 : iteraciones = 500$				
$(\alpha = 0{,}3, \beta = 0{,}4, \rho = 0{,}4,\ ants = 16)$				
C_{tot}	$t\ (años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	14,250	16	6	0,9276
	15,250	3	1	0,9730
40,0626	20,000	23	9	0,9245
	21,000	6	2	0,9633
$burning\ \ time$: 12,16 hrs				

Tabla 20*a* Mejor solución de Bitachi con tales parámetros.

$Solución\ 2 : iteraciones = 200$				
$(\alpha = 0{,}3, \beta = 0{,}35, \rho = 0{,}2,\ ants = 32)$				
C_{tot}	$t\ (años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	14,250	8	3	0,9495
	15,625	6	2	0,9297
39,2626	18,625	23	9	0,9194
	19,750	3	1	0,9260
	22,250	15	6	0,9638
$burning\ \ time$: 12,89 hrs				

Tabla 20*b* Mejor solución de Bitachi con tales parámetros.

Como puede observarse, las soluciones presentadas mejoran a las obtenidas previamente con 4 procesadores. Se amplió la búsqueda a 16 y 32 agentes. Cabe mencionar que los parámetros que se presentan en cada solución fueron aquellos con los que se obtuvo una mejora considerable. Empíricamente los valores para la terna mostrados en las soluciones fueron con los que encontraban repetidamente mejores soluciones. Puede apreciarse que ampliando el espacio de búsqueda se mejoraron las soluciones propias y con un menor número de iteraciones.

A continuación se presenta la gráfica correspondiente a la mejor solución encontrada

con Bitachi:

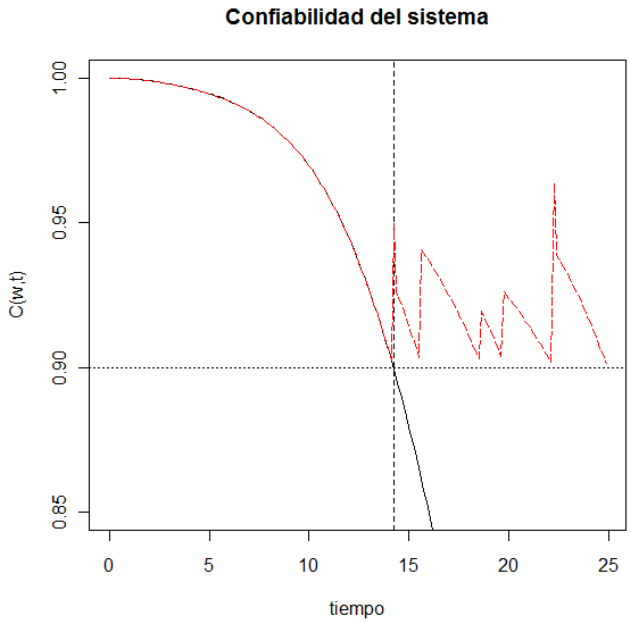

Figura 16 Confiabilidad del sistema a través del tiempo de vida del mismo.

En muchos de los resultados obtenidos, el número de iteraciones tuvo que ser mayor dependiendo de la elección de parámetros. Se muestran estas soluciones como las mejores hasta cierto momento. Se está cerca de la mejor solución en [1] pero aún no se llega a esta. Dado el tiempo y las limitaciones de este trabajo, se presentan las soluciones anteriores con posibilidades de seguir trabajando a futuro en el espacio de soluciones y su exploración; tal parece que la mejora de las soluciones se encuentra ligada a la elección de los parámetros antes mencionados.

4.2.2. Escenario 2

En este caso se consideran límites de $w = 1{,}0$ y $R' = 0{,}9$, se desea que el sistema cumpla con el 100 % de la demanda solicitada con un nivel de confiabilidad no menor a un 90 %. Para el cálculo de dicha confiabilidad considerando el factor demanda, se utilizó la función generadora universal de igual modo que el escenario anterior.

Para este caso, el tiempo al cual se debe realizar el primer mantenimiento o acción PM es $t_1 = 10{,}625$. La siguiente gráfica muestra el desempeño del sistema bajo estas

consideraciones.

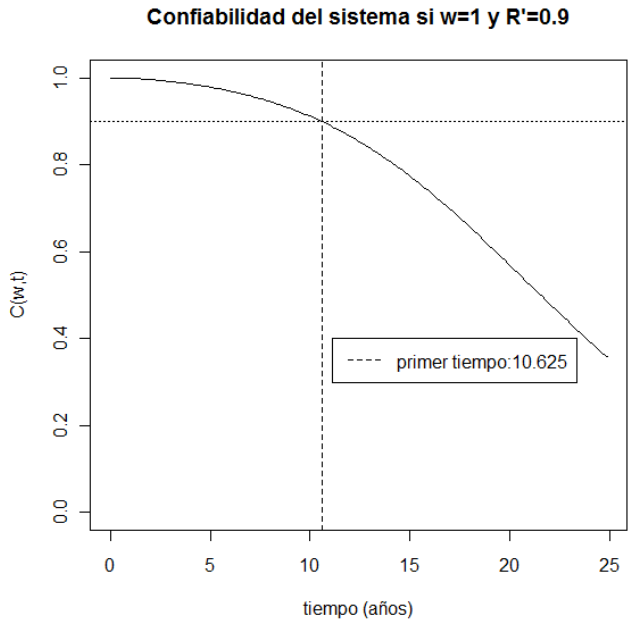

Figura 17 Desempeño del sistema considerando w y R'.

Como puede observarse el desempeño del sistema considerando la demanda presenta cambios significativos e indica el momento al cual la confiabilidad del sistema se encuentra por caer por debajo del 90 %.

Nuevamente recordamos el problema a optimizar:

$$\begin{aligned} \text{mín } C_{tot}(x^*) &= \text{mín } \{C_{PM}(x^*) + C_{MR}(x^*)\} \\ s.a.\ \ R(x^*, t, 1{,}0) &\geq 0{,}90; \quad 0 \leq t \leq 25. \end{aligned}$$

Tenemos el mismo argumento que en el escenario anterior, varias simulaciones con un número grande de iteraciones sería deseable, pero a pesar de la programación en paralelo, el tiempo de ejecución resultó ser alto. Las mejores soluciones con 4 procesadores

se muestran a continuación:

$iteraciones = 100$				
C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	10,625	8	3	0,9119
	10,875	3	1	0,9107
	11,500	5	2	0,9062
	11,625	18	7	0,9696
	11,625	13	5	0,9696
74,1626	17,250	5	2	0,9172
	17,750	11	4	0,9173
	18,750	8	3	0,9063
	19,000	15	6	0,9477
	20,625	9	3	0,9077
	21,000	18	7	0,9466
	24,125	6	2	0,9151
$burning\ \ time$: 8,46 $hrs.$				

Tabla 21*a* Mejor solución obtenida con 4 procesadores y 100 iteraciones.

$iteraciones = 500$				
C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	10,625	8	3	0,9119
	10,875	18	7	0,9617
	14,25	12	5	0,9598
	14,25	16	6	0,9598
	15,5	6	2	0,9506
68,46265	18,375	9	3	0,9179
	19,375	3	1	0,9346
	21,375	17	7	0,9563
	22,5	1	1	0,9454
	22,5	18	7	0,9454
$burning\ \ time$: 1,43 $días.$				

Tabla 21*b* Mejor solución obtenida con 4 procesadores y 500 iteraciones.

Cabe destacar que los parámetros usados en estas simulaciones fueron $\rho = 0{,}5$, $\alpha = 0{,}4$ y $\beta = 0{,}45$, el número de agentes fue 4 y 8 Incrementando el número de iteraciones como es de esperarse, se encuentra una mejor solución. Sin embargo, en este caso utilizando tan solo 4 hormigas, el tiempo de ejecución resultó bastante grande. De

esta forma se procedió como en el escenario anterior.

Dada la experiencia adquirida, se procedió en la misma computadora de 4 procesadores a correr el proceso pero utilizando 16 agentes y 100 iteraciones, también se cambiaron los valores de los parámetros por $\rho = 0{,}6$, $\alpha = 0{,}3$ y $\beta = 0{,}4$, de esta forma se le da mayor importancia al depósito de feromona que a la evaporación, más importancia a la matriz de feromonas que a la de información. Realizados dichos cambios se obtuvo

$iteraciones = 100 \quad hormigas = 16$				
C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	10,625	2	1	0,9119
	10,875	18	7	0,9617
	14,25	2	1	0,9311
	15,125	9	3	0,9275
	16,750	5	2	0,9210
67,3626	17,375	16	6	0,9412
	19,500	21	9	0,9270
	19,500	7	3	0,9270
	19,500	6	2	0,9270
	20,625	2	1	0,9197
	21,125	18	7	0,9560
	24,375	2	1	0,9232
$burning\ \ time$: 18,40 $hrs.$				

Tabla 22 Solución encontrada mejorando los parámetros de ACO.

La solución obtenida es mejor que las anteriores utilizando más hormigas y con cambio de parámetros.

Se tiene a continuación la mejor solución encontrada en [1] para este escenario:

C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	10,625	18	7	0,956
	13,625	3	1	0,939
	16,000	15	6	0,934
51,301	17,625	6	2	0,925
	19,000	8	3	0,930
	20,500	10	4	0,913
	21,250	18	7	0,956
	24,375	7	3	0,915

Tabla 23 Solución presentada por Lisnianski y Levitin.

Comparando esta solución y la propia se observa que.aun no se acerca al costo mínimo reportado, es por esto que se procedió a agilizar el proceso utilizando de igual forma Bitachi.

Algunos de los resultados que se obtuvieron incrementando el número de hormigas y haciendo variaciones en los parámetros son:

$Solución\ 1 : iteraciones = 200$				
$(\alpha = 0{,}3, \beta = 0{,}4, \rho = 0{,}4,\ ants = 16)$				
C_{tot}	$t\ (años)$	$acciones\ PM$	$elemento$	$R_p\,(t,w)$
	10,625	2	1	0,911
	10,875	11	4	0,913
64,0626	11,625	18	7	0,968
	15,75	9	3	0,929
	17,375	1	1	0,925
	17,375	5	2	0,925
	17,875	16	6	0,944
	20,125	2	1	0,924
	20,750	18	7	0,945
	23,125	6	2	0,934
$burning\ \ time : 16{,}08\ hrs$				

Tabla 24*a* Mejor solución de Bitachi para el escenario 2 con ciertos parámetros.

$Solución\ 2 : iteraciones = 200$				
$(\alpha = 0{,}2, \beta = 0{,}45, \rho = 0{,}3,\ ants = 32)$				
C_{tot}	$t\ (años)$	$acciones\ PM$	$elemento$	$R_p\,(t,w)$
	10,625	18	7	0,955
	13,500	21	9	0,941
63,26	13,500	6	2	0,941
	15,875	9	3	0,918
	16,875	3	1	0,923
	18,375	11	4	0,905
	18,750	18	7	0,935
	21,625	21	9	0,966
	21,625	16	6	0,966
$burning\ \ time :\ 1{,}44\ días$				

Tabla 24*b* Mejor solución de Bitachi para el escenario 2 con ciertos parámetros.

Estas soluciones son las mejores obtenidas en Bitachi para este escenario. Claramente se observa que al diversificar el espacio de búsqueda mediante el aumento del número

de hormigas, hay una mejora, tanto en el costo total como en el número de acciones a implementar.

A continuación se tiene la gráfica correspondiente a la mejor solución encontrada:

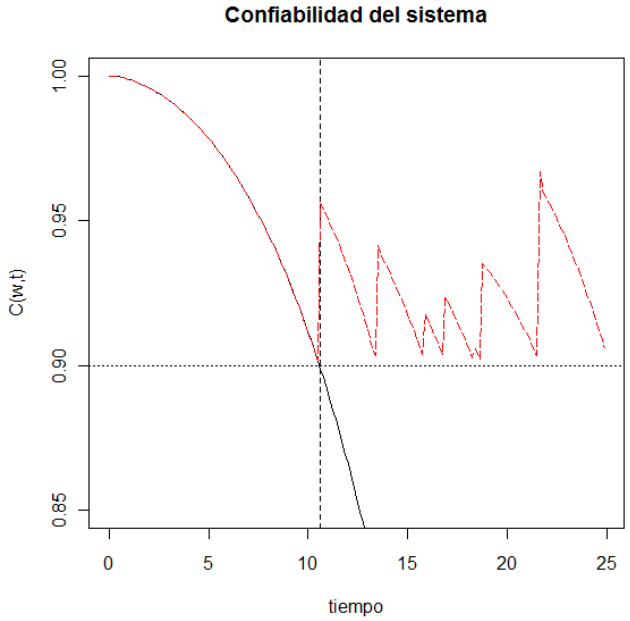

Figura 18 Desempeño de la confiabilidad dados los mantenimientos

Por otra parte se tiene la mejor solución encontrada por Lisnianski y Levitin [1]:

C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	10,625	18	7	0,956
	13,625	3	1	0,939
	16,000	15	6	0,934
51,301	17,625	6	2	0,925
	19,000	8	3	0,930
	20,500	10	4	0,913
	21,250	18	7	0,956
	24,375	7	3	0,915

Tabla 25 Solución de Lisnianski y Levitin en [1].

Las soluciones obtenidas no mejoran la anterior, pero sí se observó a lo largo del proceso que se fueron mejorando las propias contínuamente. Sin embargo, las encontradas con Bitachi poseen otra característica, en términos generales el nivel de confianza que adquieren después de cierto mantenimiento, es más alto que la solución de los autores.

4.2.3. Escenario 3

En este caso se consideran límites de $w = 1{,}0$ y $R' = 0{,}95$, se desea que el sistema cumpla con el 100 % de la demanda solicitada con un nivel de confiabilidad no menor a un 95 %. Para este caso, el tiempo al cual se debe realizar el primer mantenimiento o acción PM es $t_1 = 7{,}625$. La siguiente gráfica muestra el desempeño del sistema bajo estas consideraciones.

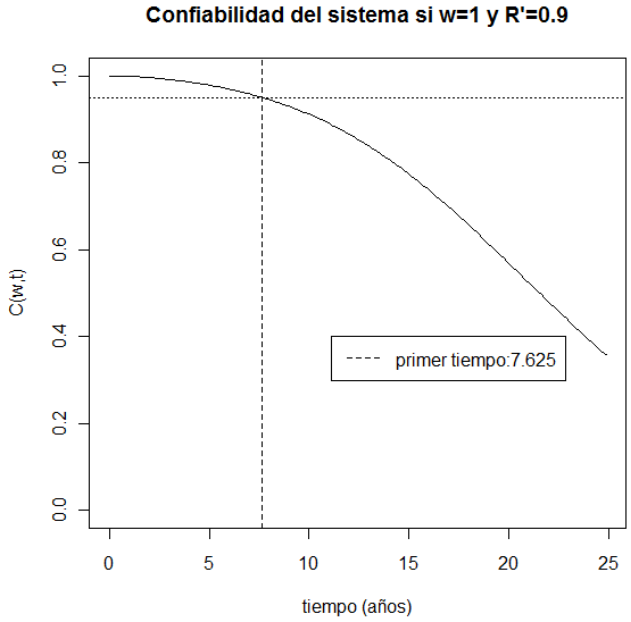

Figura 19 Desempeño del sistema en el escenario 3.

Como puede observarse, el desempeño del sistema considerando la demanda dada, presenta cambios significativos e indica el momento al cual la confiabilidad del sistema se encuentra por caer por debajo del 95 %.

Así, el problema a optimizar que tenemos es:

$$\begin{aligned} \text{mín } C_{tot}(x^*) &= \text{mín } \{C_{PM}(x^*) + C_{MR}(x^*)\} \\ s.a.\ \ R(x^*, t, 1{,}0) &\geq 0{,}95; \quad 0 \leq t \leq 25. \end{aligned}$$

Consideramos nuevamente el número de iteraciones para las cuales se ha encontrado una mejor solución, pues de igual modo, a pesar de haber programado en modo paralelo con 4 procesadores, el tiempo de ejecución resultó ser alto. Considerando que este escenario posee las restricciones más exigentes, el tiempo de ejecución para obtener una solución se incrementó considerablemente.

La siguiente solución es la mejor obtenida con 4 procesadores y $\rho = 0{,}5$, $\alpha = 0{,}4$ y

$\beta = 0{,}45$.

$iteraciones = 100$				
C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	7,625	4	2	0,9875
	7,625	7	3	0,9875
	7,625	18	7	0,9875
	10,5	1	1	0,9707
	10,5	18	7	0,9707
	11	15	6	0,9692
	11,75	9	3	0,9731
	13,625	17	7	0,9579
	13,875	8	3	0,9525
	14,000	10	4	0,9658
	14,125	3	1	0,9625
	15,125	14	5	0,9625
159,96	16,500	19	8	0,9733
	16,500	16	6	0,9733
	18,375	22	9	0,9531
	18,500	28	11	0,9628
	19,000	21	9	0,9927
	19,000	4	2	0,9927
	19,000	7	3	0,9927
	19,000	18	7	0,9927
	21,750	11	4	0,9742
	23,500	4	2	0,9759
	23,500	7	3	0,9759
	23,500	18	7	0,9759
	24,375	1	1	0,9766
	24,375	28	11	0,9766
$burning\ \ time$: 19,73 $hrs.$				

Tabla 26 Mejor solución utilizando 4 procesadores.

Para este caso se tiene la siguiente gráfica del comportamiento de la confiabilidad a través del tiempo de vida del sistema. Se puede observar el cómo aumenta en cada tiempo marcado evitando así que baje del nivel R' dado. Por otra parte nótese que el tiempo de ejecución fue de un poco más de 19 hrs; resulta ser mucho tiempo. Esto puede deberse a que el escenario es más exigente y por lo tanto hay muchas soluciones que no son factibles, es decir, para construir una factible se requiere mucho más tiempo

de ejecución.

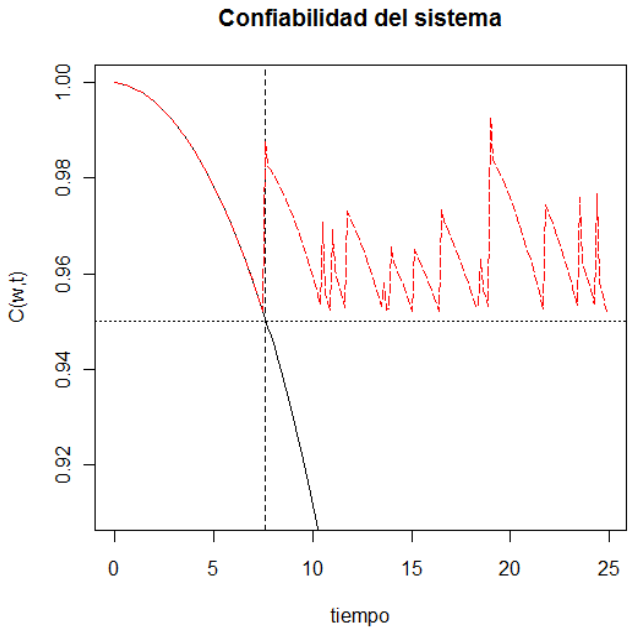

Figura 20 Solución obtenida bajo 4 procesadores.

La línea punteada corresponde a los incrementos de la confiabilidad después de haber aplicado algún mantenimiento en ciertos elementos.

Se presentarán a continuación las mejores soluciones encontradas con Bitachi.

$Solución\ 1 : iteraciones = 400$				
$(\alpha = 0{,}4, \beta = 0{,}45, \rho = 0{,}5,\ ants = 32)$				
C_{tot}	$t\ (años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	7,625	16	6	0,960
	8,375	1	1	0,957
	8,375	8	3	0,957
	8,625	18	7	0,991
	12,250	7	3	0,971
	12,250	16	6	0,971
116,86	12,500	7	3	0,978
	12,500	18	7	0,978
	13,250	7	3	0,999
	13,250	16	6	0,999
	13,250	23	9	0,999
	13,250	12	5	0,999
	13,250	27	11	0,999
	13,250	18	7	0,999
$burning\ \ time : 4{,}22\ días$				

Tabla 27a Mejor solución de Bitachi variando parámetros e iteraciones.

$Solución\ 2 : iteraciones = 100$				
$(\alpha = 0{,}4, \beta = 0{,}099, \rho = 0{,}35,\ ants = 32)$				
C_{tot}	$t\ (años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	7,625	3	1	0,953
	7,875	18	7	0,985
	11,500	16	6	0,972
	13,000	2	1	0,959
	13,375	18	7	0,968
	14,750	29	11	0,956
100,26	15,125	13	5	0,982
	16,125	9	3	0,976
	18,250	11	4	0,968
	19,625	18	7	0,973
	21,500	6	2	0,968
	23,000	3	1	0,958
	23,750	18	7	0,960
$burning\ \ time :\ 1{,}12\ días$				

Tabla 27b Mejor solución de Bitachi variando parámetros e iteraciones.

Se observa que simplemente con la modificación de los parámetros de ACO, es suficiente para mejorar el número de iteraciones y la solución obtenida.

El tiempo de ejecución se redujo considerablemente en los 16 procesadores, sin embargo, con 400 iteraciones y 32 hormigas resultó ser de 4 días, lo cual resulta aún demasiado. La siguiente gráfica corresponde a la mejor solución encontrada en Bitachi:

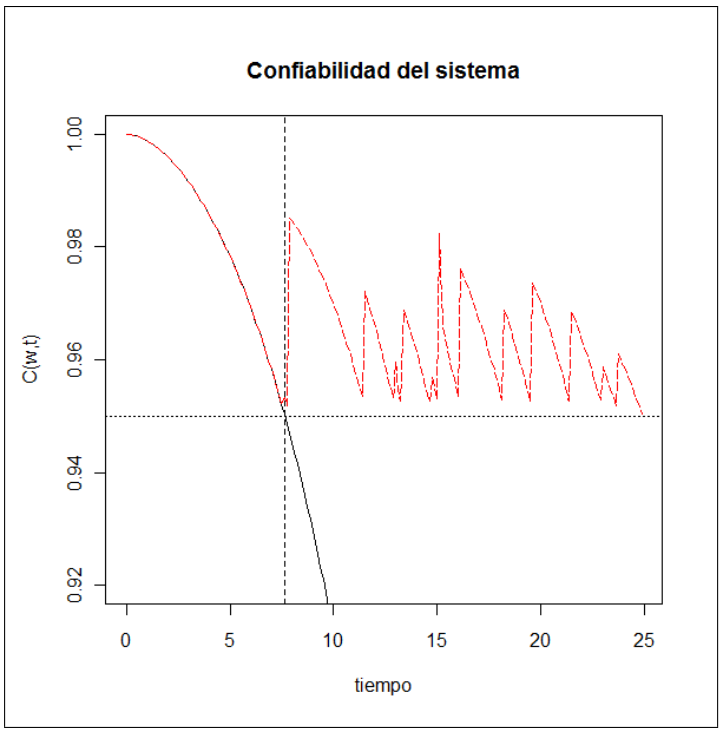

Figura 21 Mejor solución obtenida con Bitachi para el escenario 3.

La mejor solución encontrada por Lisnianski y Levitin [1] para este caso resulta:

C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	7,750	18	7	0,982
	10,750	3	1	0,963
	11,875	10	4	0,959
	12,625	18	7	0,964
	14,000	16	6	0,978
	16,125	6	2	0,969
82,625	17,625	9	3	0,965
	18,875	3	1	0,955
	19,375	27	11	0,963
	20,375	18	7	0,983
	23,125	10	4	0,965
	24,250	17	7	0,958
	24,750	4	2	0,956

Tabla 28 Solución de Lisnianski y Levitin para el escenario 3.

comparando esta solución y la propia claramente se observa que aún se encuentra lejos de dicho costo.

4.2.4. Escenario 4

En este caso se consideran las restricciones de $w = 0{,}8$ y $R' = 0{,}95$. Se desea que el sistema cumpla con el 80 % de la demanda solicitada con un nivel de confiabilidad no menor a un 95 %.

Para este caso, el tiempo al cual se debe realizar el primer mantenimiento o acción PM es $t_1 = 11{,}625$. La siguiente gráfica muestra el desempeño del sistema bajo estas

consideraciones.

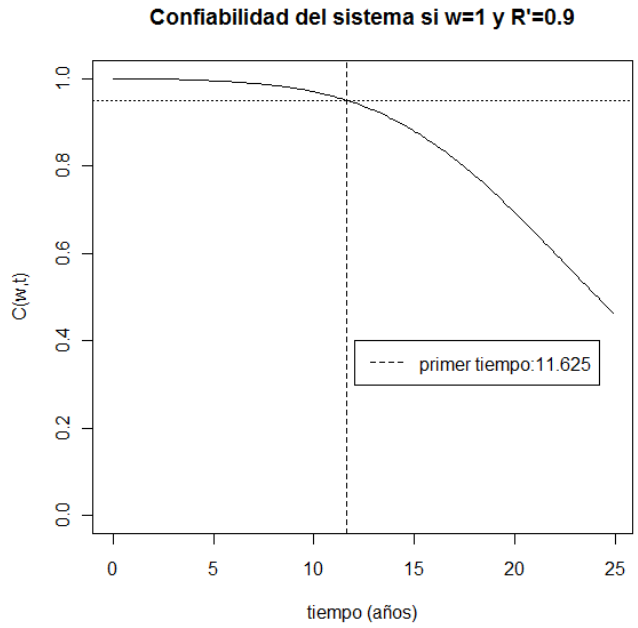

Figura 22 Confiabilidad del sistema bajo el escenario 4.

Así, el problema a optimizar que tenemos es:

$$\begin{aligned} \text{mín } C_{tot}(x^*) &= \text{mín } \{C_{PM}(x^*) + C_{MR}(x^*)\} \\ s.a.\ \ R(x^*, t, 0{,}8) &\geq 0{,}95; \quad 0 \leq t \leq 25. \end{aligned}$$

Las siguientes soluciones son las mejores obtenidas utilizando 4 procesadores. Para la

primera se utilizaron $\rho = 0{,}5$, $\alpha = 0{,}4$ y $\beta = 0{,}45$ con 4 agentes de búsqueda.

$iteraciones = 100$				
C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	11,625	23	9	0,9537
	11,875	11	4	0,9724
	14,125	6	2	0,9620
	15,750	21	9	0,9529
72,8626	15,750	17	7	0,9529
	15,875	16	6	0,9866
	19,500	19	8	0,9785
	19,500	9	3	0,9785
	22,500	23	9	0,9566
	23,000	11	4	0,9629
	24,375	6	2	0,9547
$burning\ \ time$: 5,62 $hrs.$				

Tabla 29a Solución de 4 procesadores con 100 iteraciones.

$iteraciones = 300$				
C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	11,625	22	9	0,9537
	11,750	9	3	0,9707
	14,250	3	1	0,9584
	15,375	16	6	0,9844
62,3626	19,250	22	9	0,9647
	19,875	9	3	0,9650
	21,375	18	7	0,9644
	22,625	6	2	0,9657
	24,750	23	9	0,9537
$burning\ \ time$: 12,30 $hrs.$				

Tabla 29b Solución de 4 procesadores con 100 iteraciones.

Para el caso de las 300 iteraciones se observa que estas fueron suficientes para encontrar una mejor solución que la propuesta por los autores. La siguiente gráfica

corresponde a dicho caso.

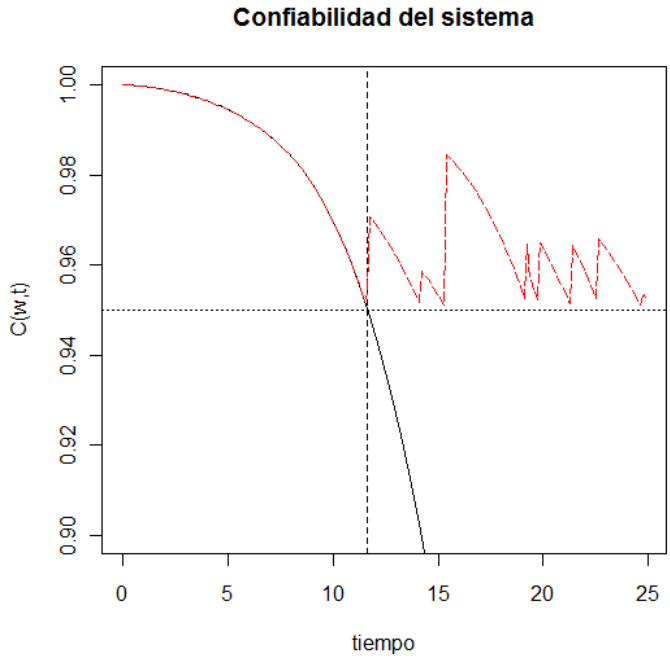

Figura 23 Desempeño del sistema con una mejor solución.

A continuación se presenta la mejor solución encontrada por Lisnianski y Levitin [1] para este escenario.

C_{tot}	t $(años)$	$acciones\ PM$	$elemento$	$R_p(t,w)$
	11,750	8	3	0,9707
	13,500	18	7	0,9578
	14,000	6	2	0,9530
	15,875	3	1	0,9820
63,669	16,500	16	6	0,9783
	21,625	9	3	0,9621
	23,125	21	9	0,9626
	24,500	1	1	0,9522

Tabla 30 Solución para el escenario 4 utilizando 4 procesadores.

Como puede observarse, la aplicación de ACO mejoró la solución encontrada por el autor en el escenario 4. Se sospecha que aumentando el número de iteraciones probablemente se encuentre aún una mejor solución. Desafortunadamente el tiempo de ejecución no fue siempre el deseado, sin embargo, con las iteraciones mostradas bastó para obtener soluciones y mejorarlas. Cabe mencionar que en este escenario no se

utilizó Bitachi dado que se pudo encontrar una solución mejor. utilizando $\alpha = 0{,}3$, $\beta = 0{,}4$ y $\rho = 0{,}6$.

Capítulo 5

Conclusiones y Recomendaciones

En esta parte se hace una recapitulación del trabajo realizado. Posteriormente, una discusión sobre trabajo a futuro.

5.1. Resumen

En este trabajo se estudió la confiabilidad en sistemas multi-estado cuya estructura obedece a las de transmisión, específicamente en los dos casos abordados se tiene MSS en serie-paralelo. Algunas de las contribuciones son:

- Explicación detallada de la metodología para poder aplicar cualquier algoritmo o procedimiento de optimización a estos problemas en particular. Cabe destacar que no solamente es factible para estos dos casos, puede servir como una guía para cualquier otro problema de un sistema multi-estado con estructura serie-paralelo.

- Aplicación del algoritmo de Colonia de Hormigas a problemas de mantenimiento perfecto e imperfecto en sistemas multi-estado.

- Mostrar la facilidad de programar en paralelo en R [7].

5.2. Conclusiones

Algunos de los problemas que se presentaron durante el desarrollo del trabajo fueron el poder entender y establecer la metodología a aplicar. Cuando se tomo el texto de Lisnianski y Levitin solamente se tenía el problema descrito en base al Algoritmo Genético. En dicho texto se explican todas las consideraciones y funciones propias de dicho algoritmo y un bosquejo del pseudocódigo que se sugiere seguir.

Con respecto a este punto, el texto es poco explícito en el cómo aplicar la función de estructura para el cálculo y re-cálculo del desempeño del sistema después de haber asignado un cierto mantenimiento a la secuencia deseada. Lo mismo sucede con la función generadora universal (UGF). No se tiene una expresión dada para el cálculo de la confiabilidad MSS dado un cierto nivel de demanda. Dados estos puntos, la contribución importante con respecto a la metodología pues es lo suficientemente explícita y clara con respecto a las mostradas en textos y artículos.

Al implementar ACO como algoritmo en la parte de optimización, el problema u obstáculo principal era el cómo utilizar la estocasticidad de este mismo; pues al poseer una matríz P que debe ser de naturaleza Markoviana, el algoritmo debe ayudar a encontrar una solución factible y óptima al paso de las iteraciones. Fue así como la propuesta principal en este aspecto es el cómo construir dicha matriz P y las asociadas a la información y a las feromonas. Esto toma el papel de contribución pues dentro de la literatura utilizada y encontrada, no se halló una aplicación a problemas de mantenimiento imperfecto. La literatura que utiliza.ACO simplemente menciona lo que ya es sabido y conocido de este algoritmo de optimización. Finalmente resultó una opción últil y versatil para combinar la aplicación de las diferentes funciondes de confiabilidad y estructura.

La función generadora universal fué un instrumento muy importante en el desarrollo de este trabajo, pues se aprovecho el hecho que puede ser utilizada en una gama importante de problemas. Ofrece una alternativa completa y genérica de poder obtener información acerca del comportamiento de cierto fenómeno si es que no es fácil de obtener mediante las funciones que comunmente se manejan (función generadora de momentos, función generadora de probabilidad, etc.).

En la parte de programación en paralelo, se pudo utilizar y comprobar la facilidad para aplicar dicha herramienta. La cuestión radicaba en que como los procesos que se programaron son muy pesados, era necesario fraccionar estos mismos o poder hacer varios a la vez para ahorrar tiempo y poder obtener algunas de las ejecuciones completas del programa por las siguientes razones:

1. **1.** se debía comprobar que efectivamente ACO puede manejar este tipo de problemas,
   2. la programación de la función generadora universal y de la función de estructura era correcta,

3. poder diversificar mucho más la búsqueda en el espacio de soluciones por agente.

5.3. Trabajo a futuro

La confiabilidad de sistemas multi-estado es un tema que parace cada vez adquirir una mayor relevancia tan solo considerando el aspecto competitivo de las empresas hoy en día. El poder mantener sistemas funcionando y en buen estado es un estado que muchas industrias desearían lograr. En la vida real el lograr una política de mantenimientos pareciera utópica. Sin embargo, el acercamiento a la realidad que nos ofrecen estos modelos no dista de un escenario que ayude a proponer soluciones y alternativas a problemas cotidianos.

Es por ello que una de las líneas de investigación a futuro podría ser el realizar investigación acerca del riesgo asociado a mantener siempre confiable un sistema MSS.

En cuanto a la aplicación de ACO, dado que resultó flexible al problema, tomando en consideración el hecho de que posee parámetros que son determinados por el usuario, el poder establecer criterios del cómo seleccionar dichos parámetros sería un estudio a futuro en la línea de dicho algoritmo. Poder realizar simulaciones con un problema mucho más sencillo y proponer los distintos escenarios bajo los cuales se decide qué combinación de parámetros usar.

En cuanto a la programación en paralelo y el programa tan pesado que se desarrolló, se puede seguir proponiendo mejoras en cuanto a la parelización, es decir, continuar investigando la forma de aplicar clusters externos y estudiar el qué tanto mejora la velocidad de procesamiento y por cada cuántos procesadores sucede esto.

La aplicación de estos modelos ha permitido estudiar fenómenos del mundo real. La confiabilidad no solo en estos casos, resulta ser un campo en el cual vale la pena invertir e investigar. Cabe aclarar que la discrepancia que existe entre los modelos reales y los teóricos siempre existirá, pero puede intentar ser reducida.

Capítulo 6

Apéndice A: Programación en Paralelo

La programación en paralelo resulta ser una herramienta bastante útil en problemas de ejecución tardada y complicada. Resulta útil en el caso de tener un proceso que requiera ser procesado en varias ocasiones de forma independiente. En este caso, el tener agentes corriendo un proceso de forma individual e independiente, permitió la fácil aplicación de esta herramienta. Muchos análisis estadísticos requieren de cómputo intesivo para poder obtener soluciones y concluir o inferir algún comportamiento. Es por ello que surge la programación en paralelo en softwafe y parquetes estadísticos como R [7].

Este tipo de programación permite reducir el tiempo de ejecución de forma importante y eficientar los recursos computacionales que se tengan, ya sean diversas computadoras, clusters o múltiples procesadores. En este caso, se utiliza la herramienta sfCluster mediante los paquetes snowfall y snow para R [7]. Estos hacen mucho más sencillo y flexible el poder realizar la parelización de algún proceso. Además de poder programar en paralelo, con dichos paquetes es posible seguir trabajando de modo secuencial.

Los procesos o los cálculos que son paralelizados, son distribuidos en agentes predefinidos para diferentes elementos de una lista. Esta herramienta puede ser utilizada al realizar bootstraping o simulaciones independientes que no necesiten ser secuenciales. En el caso de usar un cluster, este contituye un conjunto de máquinas simples llamadas nodos. De dichos nodos se eligen algunos que realizaran los procesos indicados del universo de nodos. Los cálculos son iniciados en un nodo maestro, este nodo expande las tareas paralelizadas a los demás (comunmente se les llama esclavos).

A continuación se presenta un pseudo-código en el cual se puede apreciar la estructura necesaria para paralelizar un proceso en R [7]:

```
#Inicialización de snowfall
library(snowfall)
sfInit(parallel=true, cpus=n, type="SOCK")  #el número de cpus depende del número de máquinas o procesadores que se tengan.

#Cargar los datos a utilizar o realizar el proceso previo a paralelizar
datos<-read.table(ç:/ ...")
datos<-as.matrix(datos)
data(lirios)
for(i in 1:n){
    A[,i]<-sum(exp(datos[,i]))
}

#definir la función que será paralelizada en el proceso
funcion1<-function(idx){
    do something(A[,i])
    return(X)
}
#Exportar los datos necesarios para que pueda llevarse a cabo la función
sfExport(lirios)

#aplicar la función de forma independiente en cada cpu para realizar el proceso en paralelo
resultado<-sfLapply(1:000,funcion1)

sfStop()
```

Cabe señalar que los resultados obtenidos de aplicar la función en paralelo son obtenidos en forma de lista. Esto implica que la función que se utilice en sfLapply debe estar programada de tal forma que al conjuntar en una lista los resultados de cada cpu según el número de veces que sea hecho, sea acorde a lo que se desea obtener, ya sea otra lista, una matriz de cualquier número de dimensiones, un cubo o cualquier otro modo de leer los resultados. Por ejemplo, si la información que se desea obtener al final de la paralelización debe ser una matriz, entonces la función debe ser programada para retornar un vector, de tal modo que la lista obtenida posteriormente pueda conjuntar los vectores y formar la matriz deseada.

En el caso de la información que se necesita exportar, se refiere a todas aquellas variables que sean utilizadas de forma global en la programación o que requiera la función para realizar las tareas deseadas. No deben importarse aquellas variables que

se utilicen solo internamente o se declaren dentro de la función, esto como una recomendación, pues si pueden ser exportadas pero deben ser removidas al término de su uso. El exportar hace que los objetos a ser utilizados sean inicializados en cada uno de los esclavos.

El realizar programación en paralelo también tiene algunos puntos en contra, sobre todo cuando se utiliza un cluster o varios procesadores a la vez. Dado que la computadora utiliza una buena parte de su capacidad o memoria RAM, no quedan muchos recursos para otros procesos o ejecución de programas o de degrada el desempeño de una o varias computadoras, sin embargo, esta herramienta resulta ser bastante útil en cuanto al tiempo que se reduce en la ejecución de procesos.

BIBLIOGRAFÍA

[1] Lisnianski A. y Levitin G. **Multi-State System Reliability**. World Scientific, 2003.

[2] Elmakias D. **New Computational Methods in Power System Reliability**. Springer, 2008.

[3] Grimmett G, Stirzaker D. **Probability and random processes**. Second edition. Clarendon Press, Oxford, 1992.

[4] Knaus J. et al. *Easier Parallel Computing in R with snowfall and sfCluster*. The R Journal, 1, mayo 2009.

[5] Levitin G. **The Universal Generating Function in Reliability analysis and Optimization**. Springer, 2005.

[6] Ouiddir et al., *Ant Colony Optimization for new redesign problem of multi-state*. Electrical Power System, Journal of Electrical Engineering , 55, 57-63.

[7] R Development Core Team (2010). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

[8] Ross S. **Introduction to Probability Models**. Seventh Edition. Boston: Academic Press, 2000.

[9] Snowfall manual, Package "snowfall"para R. Abril 28, 2010.

[10] Zeblah (2008), *Reliability Maximization of Power System using Ant Colony Approach*, Journal of Electrical and Power Engineering, 2(3), 192-201.