PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

# Desarrollo de Algoritmos Genéticos para la Asignación de Espacios en un Patio de Contenedores

CHRISTOPHER O'SHEE CAMPILLAY

INFORME FINAL DE PROYECTO
PARA OPTAR AL TÍTULO PROFESIONAL DE
INGENIERO DE EJECUCIÓN EN INFORMÁTICA

Julio, 2014

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

# Desarrollo de Algoritmos Genéticos para la Asignación de Espacios en un Patio de Contenedores

CHRISTOPHER O'SHEE CAMPILLAY

Profesor Guía: Claudio Cubillos Figueroa

Profesor Correferente: Rosa González Ramírez

Carrera: Ingeniería de Ejecución en Informática

Julio, 2014

*A mi esposa Natalie*

# Índice

# Resumen

El Problema de asignación de espacio en un patio de contenedores es un problema de la vida real, enfrentado por distintos puertos del mundo. Existen distintos objetivos que se desean lograr por parte de los puertos. Uno de ellos es entregar un buen servicio al minimizar el tiempo de permanencia de los barcos. Otro tiene que ver con los costos operativos. Este proyecto toma ambos aspectos. La dificultad de este problema impulsa la aplicación de heurísticas para su resolución. El objetivo de este proyecto es el desarrollo y evaluación de un algoritmo genético para resolver el problema de asignación de espacios.

*Palabras-claves: Algoritmo Genético, Problema de asignación de patio de contenedores.*

# Abstract

The *Container Yard Allocation Problem* is a real-life problem, faced by different ports in the world. Ports have diverse objectives to achieve. One of them is to deliver a good service, minimizing vessel dwell times. Another one is related to operative costs. This project considers both of them. The difficulty of this problem leads to the application of heuristics to solve it. The objective of this project is the development and evaluation of a genetic algorithm to solve CYAP.

*Keywords: Genetic Algorithm, Container Yard Allocation Problem.*

# Lista de Figuras

# 1 Introducción

Alrededor del mundo los puertos marítimos juegan un papel importante en el comercio global. Los medio de transporte marítimos proveen un eficiente modo de transportar distintas especies. Chile, por sus peculiares características geográficas, posee un borde costero muy extenso. Por lo que no es extraño encontrar la importante cantidad de 56 puertos a lo largo del país. A medida que pasa el tiempo, muchos puertos han sido arrinconados por las ciudades que se han desarrollado alrededor. Esto impide la posibilidad de ampliar el terreno destinado al uso por parte de los puertos para desarrollar sus actividades. La demanda por los servicios de un puerto, al incrementarse, provoca que los puertos busquen una manera de aprovechar mejor sus espacios, ante la imposibilidad de ampliar sus explanadas hacia tierra y el elevado costo que tiene crear terrenos hacia el mar.

Sin embargo, no es lo único que buscan los puertos en general. Otro aspecto es la calidad de servicio que ellos proveen y la disminución de sus costos operativos, por lo que este proyecto considera varios aspectos. Se verá que el problema no es fácil de resolver, y que heurísticas son necesarias para resolverlo en un tiempo razonable. En este caso se propone una solución mediante Algoritmos Genéticos, que son heurísticas que imitan los mecanismos de la selección natural, donde cada solución es un individuo. Estos individuos ponen a prueba su aptitud para sobrevivir, por lo tanto, las mejores soluciones sobreviven. Luego, las soluciones se reproducen, heredando sus genes a la siguiente generación. En Teoría, después de varias generaciones, el algoritmo converge y entrega la solución óptima, o la mejor que pudo encontrar.

## 1.1 Objetivos del Proyecto

### 1.1.1 Objetivo General

- Desarrollar algoritmos genéticos para resolver el problema de asignación de espacios en un patio de contenedores.

### 1.1.2 Objetivos específicos

Investigar cómo afectan los distintos parámetros y operadores en el rendimiento de un algoritmo genético.

Estudiar distintas formulaciones del problema y analizar la posibilidad de ser resuelto con algoritmos genéticos.

- Evaluar mediante pruebas reales el rendimiento de todos los algoritmos propuestos, y compararlos, de ser posible, con otras publicaciones.

## 1.2 Plan de Trabajo y metodología

El objetivo del proyecto no es el desarrollo de un sistema de información, por lo que no se usará una metodología típica de desarrollo de software. Aunque al final del proyecto se entregará un software funcional, lo más importante son los resultados que este entrega, no el software en sí. No obstante el desarrollo de los distintos algoritmos genéticos se usará una metodología iterativa incremental, implementando los algoritmos en paralelo. A excepción de un primer prototipo que se presenta en el presente informe.

El trabajo se divide en 4 partes, las que se explicarán a continuación en detalle.

### 1.2.1 Primera etapa: Investigación y definición del problema

En esta etapa se revisó la literatura existente en el ámbito del problema a resolver y la técnica que se utilizará para resolver el problema. Luego etapa se propuso una definición tentativa del problema.

Duración: 2 meses (Marzo - Abril, 2012).

### 1.2.2 Segunda etapa: Diseño e implementación de un primer prototipo

En el transcurso de la etapa se experimentaron con las herramientas disponibles. Se investigaron y probaron distintas opciones para implementar el algoritmo genético y se tomaron las decisiones pertinentes. Al final de esta etapa se modificó completamente la definición del problema.

Duración: 2 meses (Mayo - Junio, 2012).

### 1.2.3 Tercera etapa: Implementación de algoritmos genéticos

En base al resultado de la etapa anterior se implementó el algoritmo genético con todas las características propuestas sobre el modelo final. Luego se realizaron ajustes menores al modelo. Sobre este modelo se efectuaron pruebas con datos ficticios.

Duración: 2,5 meses (Agosto - 1° mitad de Octubre, 2012).

### 1.2.4 Cuarta etapa: Ajustes y pruebas finales.

En esta etapa se efectuaron los ajustes finales del modelo. Se efectuaron las pruebas finales con datos reales. Durante las pruebas se corrigieron algunos errores que ocurrían solo bajo ciertos escenarios.

Duración: 1,5 meses (Marzo - Abril, 2013).

## 1.3 Estructura del Documento

En el capítulo uno se expone la introducción al presente informe, luego están el objetivo general y los objetivos específicos del proyecto de título del autor, para terminar con la metodología utilizada y el plan de trabajo.

El capítulo dos trata sobre los algoritmos genéticos, la relación que tienen con la naturaleza, el modelo simple de Holland, fundamentos teóricos y alternativas al modelo de Holland. Esto, para comprender la utilidad de esta heurística para problemas de optimización y comprender la propuesta de solución al problema.

En el capítulo tres se expone una idea general de las operaciones de un puerto. Más adelante se muestran trabajos anteriores relacionados del ámbito de las operaciones en un

patio de contenedores comentando sus resultados y conclusiones, en especial aquellos que utilizan la técnica heurística a utilizar en este trabajo.

En el capítulo cuatro se muestra la definición formal del problema a resolver. Se describe el escenario de estudio. En la segunda parte de este capítulo se explican los operadores genéticos a utilizar. Terminando con el diseño de la implementación del algoritmo.

El capítulo cinco contiene una descripción de las pruebas realizadas en el marco del proyecto. Se incluyen las pruebas de sensibilidad de los parámetros del algoritmo genético y las pruebas finales con datos reales.

Para terminar, en el capítulo seis se encuentran las conclusiones obtenidas hasta la fecha.

# 2 Algoritmos Genéticos

En la naturaleza, los individuos más adecuados para competir por recursos escasos sobreviven. Adaptarse a los cambios de entorno es esencial para la supervivencia de los individuos de cada especie. Mientras que las diversas características que caracterizan un individuo determinan su capacidad de supervivencia, las características de turno están determinadas por el código genético del individuo. Específicamente, cada característica está controlada por una unidad básica llamada gen. El conjunto de genes que controlan las características forman los cromosomas, los cuales son la clave para la supervivencia de un individuo en un entorno competitivo. Aunque la evolución se manifiesta como una sucesión de cambios en las características de las especies, son los cambios en el material genético de la especie que forman la esencia del a evolución. Específicamente, lo que impulsa la evolución es la articulación de la selección natural y la recombinación del material genético que ocurre que ocurre durante la reproducción.

En la naturaleza, la competencia entre individuos por recursos escasos como la comida, el espacio o parejas resultan en la dominación de los individuos más aptos sobre los más débiles. Solo los individuos más aptos sobreviven y se reproducen, un fenómeno conocido como “la supervivencia del más apto”. Por lo tanto, los genes del más apto sobreviven, mientras que los de los más débiles se pierden en la historia. La selección natural lleva a la supervivencia de los individuos más aptos, pero también implícitamente lleva a la supervivencia de los genes más aptos.

El proceso der reproducción genera diversidad en el pozo de genes. La evolución se inicia cuando el material genético de dos progenitores se recombina durante la reproducción. Nuevas combinaciones de genes son generadas de las anteriores, un nuevo pozo de genes se crea. Específicamente, el intercambio de material genético entre cromosomas es llamado *cruza* Segmentos de los cromosomas de los dos progenitores se intercambian durante el cruza, creando la posibilidad de la combinación correcta de genes para mejores individuos. La selección y cruza repetitiva causa la evolución continua del pozo de genes y de la generación de individuos que sobrevive mejor en un ambiente competitivo.

Holland propuso los algoritmos genéticos en los comienzos de la década de 1970 como programas que imitan procesos evolutivos en la naturaleza [HOL75]. Los algoritmos genéticos manipulan una población de potenciales soluciones un problema de optimización. Específicamente, estos operan sobre representaciones codificadas de las soluciones, equivalente al material genético de los individuos en la naturaleza, no sobre las soluciones mismas. El algoritmo genético de Holland codifica las soluciones como una cadena de bits provenientes de un alfabeto binario. Tal como ocurre en la naturaleza, la selección provee del

mecanismo de conducción necesaria para que las mejores soluciones sobrevivan. Cada solución es asociada a un *valor de aptitud* que refleja que tan buena es, en comparación a otras soluciones en la población. Mientras más grande sea el valor de aptitud de un individuo, más grande será la probabilidad de supervivencia y reproducción, y más grande será su representación en la generación subsiguiente. La recombinación del material genético en un algoritmo genético es simulada a través de un mecanismo de cruza que intercambia porciones entre cadenas. Otra operación, llamada mutación, causa alteraciones aleatorias y esporádicas de los bits de la cadena. La mutación también tiene una analogía directa en la naturaleza y toma el rol de regenerar material genético perdido y crear material nuevo.

## 2.1 El algoritmo de Holland

En la literatura, al algoritmo genético propuesto por Holland es comúnmente llamado algoritmo genético simple o SGA (del inglés, *simple genetic algorithm*). Para el funcionamiento de un SGA es esencial una población de cadenas binarias. Cada cadena de 1s y 0s es una versión codificada de una solución al problema de optimización. Usando operadores genéticos como la cruza y la mutación, el algoritmo crea generaciones subsiguientes de las cadenas de la población actual. El ciclo generacional es repetido hasta que un criterio de término se cumpla. A continuación se encuentra la estructura básica del algoritmo genético clásico, el que resume su funcionamiento, el cual contiene los siguientes componentes: una población de cadenas binarias, parámetros de control, una función de aptitud, operadores genéticos, un mecanismo de selección y un mecanismo para codificar las soluciones como cadenas binarias.

```
Algoritmo Genético Clásico ()
1     Iniciar población
2     Evaluar población
3     Mientras criterio de término no se alcance.
4           Seleccionar soluciones para siguiente población
5           Realizar cruzamiento y mutación
6           Evaluar población
7     Fin
```

### 2.1.1 Mecanismo de codificación

El mecanismo de codificación de fundamental para la estructura de un algoritmo genético para representar las variables del problema de optimización. Este mecanismo depende en la naturaleza de las variables del problema. Por ejemplo, para obtener los flujos óptimos en un problema de transporte, las variables se asumen continuas, mientras las

variables en el TSP (problema del vendedor viajero, del inglés, *traveling salesman problem*) son cantidades binarias representando la inclusión o exclusión de una arista (*edge*) en el circuito Hamiltoniano, en cada caso el mecanismo de codificación debe mapear cada solución en una cadena binaria única.

Un gran número de problemas de optimización poseen variables continuas. Un método común de codificarlas es usar una representación entera. Cada variable es primero mapeada linealmente a un entero definido en un rango, luego este es codificado usando un número de bits fijos. Por ejemplo, considere una variable continua definida en el rango [-1.23, 1.23]. Podríamos codificar esta variable con una precisión de dos decimales multiplicando su valor real por 100, luego descartando la porción decimal del producto. Así el valor de la variable alcanza su mapeo lineal a enteros en el intervalo [-128, 128]. El código binario correspondiente a cada entero puede ser fácilmente calculado.

### 2.1.2 La función de aptitud

La función objetivo, la función a optimizar, provee el mecanismo para evaluar cada cadena. Sin embargo, el rango de sus valores varía de problema en problema. Para mantener uniformidad sobre varios dominios de problemas, se usa la *función de aptitud*, para normalizar la función objetivo a un intervalo conveniente: [0, 1]. El valor normalizado de la función objetivo es la *aptitud* de la cadena, el cual es usado por el mecanismo de selección para evaluar las cadenas de la población.

### 2.1.3 Selección

La selección se basa en el mecanismo *supervivencia de los más aptos*. Las soluciones más aptas sobreviven mientras las débiles perecen. En el SGA, una cadena más apta recibe un número más elevado de descendencia y así una probabilidad más alta de sobrevivir en la siguiente generación. En el esquema *selección proporcional*, una cadena con valor $f_i$, se le asigna una descendencia asignada $f_i/\bar{f}$, donde $\bar{f}$ es el valor de aptitud promedio de la población. A una cadena con una valor de aptitud más alto que el promedio se le asigna más de una descendencia, mientras que a una cadena con un valor más bajo que el promedio se le asigna menos de una descendencia. El esquema de selección proporcional asigna fracciones de descendencia. Por lo tanto el número $f_i/\bar{f}$ representa el número de descendencia esperado de una cadena. Ya que en la última asignación algunas cadenas reciben un número más alto de descendencia que $f_i/\bar{f}$ y otras menos que $f_i/\bar{f}$, los métodos de asignación incluyen cierto componente aleatorio para remover sesgos metódicos de asignación para a cualquier conjunto

particular de cadenas. La técnica de asignación controla el alcance para que la asignación actual de descendencias a cadenas coincida el número de descendencias $f_i/\bar{f}$ esperadas.

| Individuo | Aptitud |
|---|---|
| 1 | 1.0 |
| 2 | 1.0 |
| 3 | 2.0 |
| 4 | 1.333 |
| 5 | 1.333 |
| 6 | 0.667 |
| 7 | 0.667 |

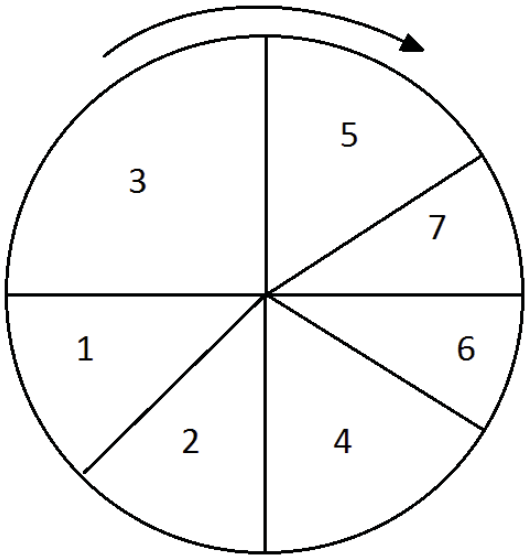


**Figura 2.1 – Selección de ruleta.** *Fuente: Elaboración propia.*

El SGA usa el esquema de *selección de ruleta* [Hol75] (figura 2.1) para implementar la selección proporcional. Cada cadena es asignada a un sector de la ruleta con un Angulo subtendido por el sector en el centro de la rueda igual a $2\pi f_i$. A una cadena se le asigna una descendencia si un numero generado al azar en el intervalo [0, $2\pi$] cae en el sector correspondiente a la cadena. El algoritmo selecciona cadenas de esta manera hasta que se ha generado la población completa de la siguiente generación. La selección de ruleta puede generar grandes errores muéstrales en el sentido que el número final de descendencia asignada a una cadena puede variar significativamente del número esperado. Los números de descendencia asignados se acercan al número esperado solo para tamaños de población muy grandes.

### 2.1.4 Mecanismo de cruzamiento

Luego de la selección viene el cruzamiento, operación crucial de un algoritmo genético. Pares de cadenas son seleccionadas aleatoriamente de la población sujeta al cruzamiento. El SGA usa la aproximación más simple: el cruzamiento de un punto (*single point crossover*). Sea ***l***, la longitud de la cadena, se elige aleatoriamente un punto de cruza que puede asumir valores en el intervalo [1, ***l*** - 1]. Las porciones de las dos cadenas después del

punto de cruza se intercambian para formar dos nuevas cadenas. El punto de cruza puede asumir cualquiera de los $\boldsymbol{l} - 1$ valores posibles con igual probabilidad. Además, el cruzamiento no siempre se efectúa. Después de elegir un par de cadenas, el algoritmo invoca el cruzamiento solo si un numero generado al azar en el intervalo [0, 1] es mayor a $p_c$, la tasa de cruzamiento (este concepto también se usa como la probabilidad de cruzamiento). De otra manera la cadena permanece sin alteraciones. El valor de $p_c$ cae en el intervalo [0, 1]. En una gran población, $p_c$ nos da la fracción de cadenas cruzadas.

### 2.1.5 Mecanismo de mutación

Después del cruzamiento, cadenas son sujetas a la mutación. La mutación de un bit involucra invertirlo, es decir, cambiar un 0 por 1 o viceversa. Tal como ocurre con $p_c$, el cual controla la probabilidad del cruzamiento, otro parámetro, la tasa de mutación $p_m$ da la probabilidad de que un bit sea invertido. Cada bit de la cadena es independientemente mutado, esto es, la mutación de un bit no afecta la probabilidad de mutación de otros bits. En un SGA, la mutación toma un papel secundario, con el objetivo de recuperar material genético perdido. Por ejemplo, suponga que todas las cadenas en una población convergen a cero en cierta posición y que la solución óptima tiene un 1 en esa posición. Entonces el cruzamiento no puede regenerar ese 1 en esa posición, mientras que la mutación si puede.

### 2.1.6 Criterio de término

Para terminar la ejecución de un SGA, se debe especificar un criterio de término. Se puede especificar que termine luego de pasado cierta cantidad de generaciones. Otra alternativa es el término al momento en que se encuentra una cadena sobre un valor de aptitud predeterminado. También la ejecución podría ser terminada cuando la población alcance cierto nivel de homogeneidad, es decir gran parte de las cadenas tiene la mayoría de los bits idénticos. Se deben usar criterios de términos ya que al ser una heurística, no es el objetivo de los GA encontrar óptimos, sino encontrar una buena solución en un tiempo razonable. En el caso de ser posible encontrar una solución óptima, en la mayoría de los casos no se puede determinar si esta se encontró.

### 2.1.7 Un ejemplo de ciclo generacional

Población P1:

| Cadena | Aptitud |
|---|---|
| 0000011100 | 0.3 |
| 1000011111 | 0.6 |
| 0110101011 | 0.6 |
| 1111111011 | 0.9 |

**Figura 2.2 – Población inicial.** *Fuente: Elaboración propia.*

La figura 2.2 muestra un ciclo generacional de un algoritmo genético con una población P1 de 4 cadenas de 10 bits cada una. En el ejemplo, La función objetivo puede tomar valores en el intervalo [0, 10], que da el número de 1s en la cadena. La función de aptitud divide por 10 para normalizar la función objetivo al intervalo [0, 1]. Las cuatro cadenas tienen valores de aptitud de 0.3, 0.6, 0.6 y 0.9 respectivamente. Idealmente, el esquema de selección proporcional debería asignar valores de descendencia de 0.5, 1, 1, y 1.5 a las cadenas. Sin embargo, en este caso, la asignación final de descendencia es 0.1, 1 y 2.

Población P2: Después de la selección

| Cadena | Aptitud |
|---|---|
| 1000011111 | 0.6 |
| 0110101011 | 0.6 |
| 1111111011 | 0.9 |
| 1111111011 | 0.9 |

**Figura 2.3 – Después de la selección.** *Fuente: Elaboración propia.*

En la figura 2.3 la población P2 representa el conjunto de cadenas seleccionadas. Luego, las cadenas son emparejadas aleatoriamente para el cruzamiento. La cadena 1 se empareja con la 4 y la cadena 2 se empareja con la 3. Con una tasa de cruzamiento de 0.5, solo una de las parejas es sujeta al cruzamiento, en este caso la pareja 1-4, mientras que la otra pareja. El punto de cruzamiento cae entre el 5to y 6to bit, por lo tanto las cadenas intercambian sus valores después del bit 5.

Población P3: Después de la cruza

| Cadena | Aptitud |
|---|---|
| 100000\|11011 | 0.5 |
| 0110101011 | 0.6 |
| 1111111011 | 0.9 |
| 11111\|11111 | 1.0 |

**Figura 2.4 – Después de la cruza.** *Fuente: Elaboración propia.*

La población P3 (figura 2.4) representa el conjunto de después del cruzamiento y su mutación puede ser apreciada en la población P4. Solo dos bits de cuarenta han sido mutados, representando una tasa de mutación de 0.05.

Población P4: Después de la mutación

| Cadena | Aptitud |
|---|---|
| 1000011011 | 0.5 |
| 0110111011 | 0.7 |
| 1111111011 | 0.9 |
| 0111111111 | 0.9 |

**Figura 2.5 – Después de la mutación.** *Fuente: Elaboración propia.*

La población P4 (figura 2.5) representa la siguiente generación. El ejemplo de las figuras 2.2, 2.3, 2.4 y 2.5, es solo una ilustración. Típicamente un SGA tiene valores de tamaño de población entre 30 y 200, tasas de cruzamiento entre 0.5 y 1, y tasas de mutación entre 0.001 a 0.05. Estos parámetros se conocen como *parámetros de control*, los cuales deben ser especificadas antes de la ejecución del algoritmo. El algoritmo termina cuando se alcanza el criterio de término, el cual no viene al caso del ejemplo.

## 2.2 Fundamento teórico

En la literatura se encuentran diversas modificaciones al modelo original de Holland, generalmente modificando parámetros y operadores. Es común ver propuestas de operaciones de cruzamiento, selección y mutación, especialmente diseñados para cierto problema puntual o

cierto tipo de problema. Siempre se demuestra su efectividad, pero no hay mucha justificación del por qué estos algoritmos funcionan. Se ve que lo pragmático avanza más rápido que lo teórico. Sin embargo, el teorema del esquema de Holland y la hipótesis de bloques de construcción de Goldberg [Gol89], exponen fundamentos teóricos para los algoritmos genéticos.

Considere un esquema, que es una plantilla de similitud, el cual describe un subconjunto de cadenas con similitudes en ciertas posiciones. En otras palabras un esquema representa un subconjunto de todas las posibles cadenas que tienen los mismos bits en ciertas posiciones. Por ejemplo, considere cadenas de 5 bits. Un esquema **000 representa cadenas con 0s en las últimas tres posiciones: las cadenas 00000, 01000, 10000 y 11000. Similarmente, el esquema 1*00* representa las cadenas 10000, 10001, 11000 y 11001. Cada cadena representada por un esquema es llamada una *instancia* del esquema. Ya que el símbolo * significa que un 0 o un 1 puede tomar la posición, el esquema ***** representa a todas las posibles cadenas de 5 bits. Las *posiciones fijas* de un esquema son las posiciones de la cadena que tienen un 0 o un 1: en el esquema **000, la tercera, cuarta y quinta, son posiciones fijas. El número de posiciones fijas es su *orden*: en **000, su orden es 3. El *defining lenght* es la distancia entre las posiciones fijas más externas: en **000 es 2 y en 1*00* es 3. Cualquier cadena específica es simultáneamente una instancia de $2^l$ esquemas, donde $l$ es la longitud de la cadena. Ya que un esquema representa un subconjunto de cadenas, podemos asociar un *valor de aptitud promedio* con un esquema: la *aptitud promedio del esquema*. En una población dada, este es determinado por la aptitud promedio de las instancias del esquema. Por lo tanto, la aptitud promedio de un esquema varía con la composición de la población de una generación a otra.

Ahora considera un esquema con $k$ posiciones fijas. Existen $2^k - 1$ otros esquemas con la misma cantidad de posiciones fijas, los que pueden ser obtenidos al considerar todas las permutaciones de 0s y 1s en estas $k$ posiciones. Además, para $k$ posiciones fijas hay $2^k$ esquemas distintos que generan particiones de todas las posibles cadenas. Cada conjunto de $k$ posiciones fijas genera una *competencia de esquema*, una competencia de supervivencia entre los $2^k$ esquemas. Ya que hay $2^l$ posibles combinaciones de posiciones fijas, $2^l$ competiciones de esquemas son posibles. La ejecución de un GA así genera $2^l$ competiciones de esquemas simultáneamente. El GA simultáneamente, pero no de forma independiente, intenta resolver todas las $2^l$ competiciones de esquemas y ubicar el mejor esquema para cada conjunto de posiciones fijas.

Se puede ver búsqueda de un GA por la cadena óptima como una competencia de esquemas simultanea para aumentar el número de sus instancias en la población. Si se describe la cadena optima como una yuxtaposición de esquemas de *distancia definitoria* pequeña y de alta aptitud promedio, entonces los ganadores de las competiciones individuales de esquemas pueden potencialmente formar la cadena óptima. Esquemas con altos valores de aptitud y

bajos *defining lenght* son propiamente llamados *bloques de construcción*. La noción de que cadenas de alta aptitud pueden ser ubicadas a través del muestreo de bloques de construcción de alta aptitud, y el combinar bloques de construcción de forma efectiva, se conoce como *hipótesis de bloques de construcción*.

### 2.2.1 Hipótesis de bloques de construcción de Goldberg

Los operadores genéticos, generan, promueven y yuxtaponen bloques de construcción para formar cadenas óptimas. El cruzamiento tiende a conservar la información genética presente en las cadenas a cruzar. Así, cuando las cadenas a cruzar son similares, su capacidad de generar nuevos bloques de construcción disminuye. La mutación no es un operador que genera radicalmente nuevos bloques. La selección provee un sesgo favorable hacia los bloques con alta aptitud y asegura al aumento de su representación de generación en generación. La yuxtaposición de bloques de construcción se alcanza en el cruzamiento y esta es la piedra angular de la mecánica del GA.

La hipótesis de bloques de construcción asume que la yuxtaposición de buenos bloques produce buenas cadenas. Esto no es siempre cierto. Dependiendo de la naturaleza de la función objetivo, pueden ser generadas muy malas cadenas cuando buenos bloques se combinan. Estas funciones objetivos se llaman funciones engañosas (*deceptive functions*).

### 2.2.2 El teorema del esquema de Holland

Cuando consideramos los efectos de la selección, cruzamiento y mutación, en la frecuencia en la que las instancias de un esquema aumentan de generación en generación, se ve que la selección proporcional aumenta o disminuye el número en relación a la aptitud promedio de un esquema. Sin considerar el cruzamiento, un esquema con alta aptitud crece exponencialmente para ganar su competición. Sin embargo, una alta aptitud por sí misma no es suficiente para una tasa de crecimiento alta. Además un esquema debe tener una baja defining lenght. Ya que el cruzamiento es disruptivo, mientras más grande es el *defining lenght*, más alta es la probabilidad de que el punto de cruzamiento caiga entre las posiciones fijas y destruyendo la instancia. Así, los esquemas con alta aptitud con bajos *defining lenght* crecerán exponencialmente. Esta es la esencia del teorema del esquema, propuesto por Holland como el *teorema fundamental de algoritmos genéticos*.

La siguiente ecuación muestra la definición formal del teorema del esquema:

$$N(h,t+1) \geq \frac{N(h,t)f(h,t)}{\bar{f}(t)}\left[1 - p_c\frac{\delta(\boldsymbol{h})}{l-1} - p_m o(h)\right]$$

Dónde:

***f(h,t)***: Aptitud promedio de un esquema h en la generación ***t***.

$\bar{f}$ ***(t)***: Aptitud promedio de una población en la generación ***t***.

$p_c$: Probabilidad de cruzamiento.

$p_m$: Probabilidad de mutación.

***δ(h)***: *Defining lenght* del esquema ***h***.

***o(h)***: orden del esquema ***h***.

***N(h,t)***: Numero esperado de instancias del esquema h en la generación ***t***.

***l***: Longitud de una cadena

El factor:

$$p_c\frac{\delta(\boldsymbol{h})}{l-1}$$

Da la probabilidad de que una instancia de un esquema ***h*** sea alterada por el cruzamiento. Y $p_m o(h)$ da la probabilidad de que una instancia sea alterada por mutación.

El GA muestrea los bloques de construcción a una muy alta tasa. En un solo ciclo generacional el GA procesa solo *P* cadenas (P, tamaño de la población), pero implícitamente evalúa aproximadamente $P^3$ esquemas [GOL89]. Esta capacidad de procesar simultáneamente un gran número de esquemas es llamada *paralelismo implícito*, y viene del hecho de que una cadena simultáneamente representa $2^l$ esquemas distintos.

## 2.3 Modificaciones posteriores al algoritmo clásico de Holland

En las últimas décadas, se ha investigado como mejorar el rendimiento de los algoritmos genéticos. Se han propuesto implementaciones más eficientes de mecanismos de selección, como la selección basada en ranking, estrategias elitistas, selección *steady-state* y selección de torneo. Estas han sido propuestas como alternativas al mecanismo de selección proporcional. Así mismo, se han propuestos otros mecanismos de cruzamiento como el

cruzamiento de dos puntos, multipunto y cruzamiento uniforme como mecanismos mejorados, por sobre el cruzamiento de un punto. El código *Gray* y la codificación dinámica [Schb92] han superado algunos problemas asociados a la codificación entera. Además se han propuesto técnicas adaptativas que dinámicamente varían los parámetros de control. Otras innovaciones incluyen algoritmos genéticos distribuidos y algoritmos genéticos paralelos [Can98].

### 2.3.1 Mecanismos de selección

La selección proporcional asigna la descendencia basada en la aptitud de una cadena y la aptitud promedio de una población. En las primeras generaciones del GA, generalmente la población tiene un bajo promedio de aptitud. La presencia de pocas cadenas de relativamente buena aptitud causa que la selección proporcional asigne una gran descendencia a estas *super cadenas*, apoderándose de toda la población, causando una convergencia prematura. Luego surgen otros problemas en etapas posteriores del GA cuando la población converge y la variación en la aptitud de las cadenas se hace muy poca. A continuación se mostraran algunas alternativas al mecanismo de selección proporcional.

#### 2.3.1.1Selección por muestreo estocástico universal

De una manera muy parecida al mecanismo de selección de ruleta, los individuos son asignados a segmentos de línea contiguos, tal que el tamaño de cada segmento es equivalente a la aptitud del individuo que le corresponde, como se ve en la figura 2.6 (línea negra). En este caso una cantidad de punteros separados por un espacio se ponen sobre la línea, esta cantidad está determinada por la cantidad de individuos a seleccionar (en verde) [Jam87]. Considere una cantidad *N* de individuos a seleccionar, luego la distancia que separa a los punteros unos de otros es *1/N* y la posición del primer puntero está dado por un numero generado aleatoriamente en el intervalo [0, *1/N*] (en azul).

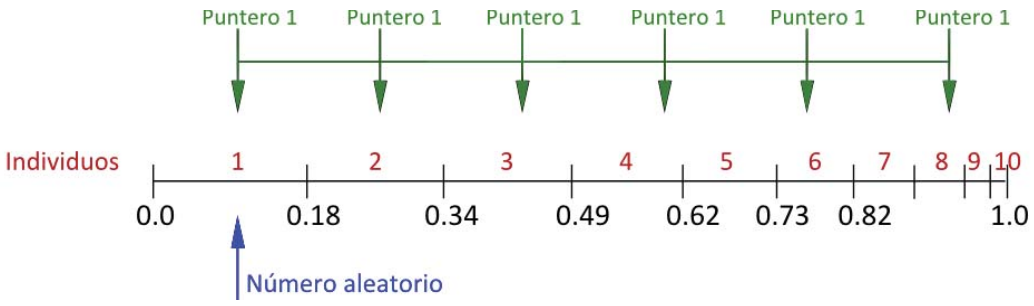


**Figura 2.6 – Selección por muestreo estocástico universal.** *Fuente: Elaboración propia.*

Para una cantidad de 6 individuos a seleccionar, la distancia entre los punteros es 1/6 = 0.167. La figura 2.6 ilustra la selección para este ejemplo. Luego de la selección, la población

destinada al cruzamiento consiste en los individuos 1, 2, 3, 4, 6 y 8. La selección por muestreo estocástico universal asegura una selección de descendencia, la cual está más cerca a lo merecido que en la selección por ruleta.

### 2.3.1.2Selección por torneo

En la figura 2.7 se muestra el mecanismo de selección por torneo, selecciona individuos (cadenas) para insertarlos en un conjunto reproductivo. Los individuos de este conjunto son usados para generar un nueva descendencia, cuyo resultado forma la base del a siguiente generación. Ya que individuos en este conjunto son los que cuyos genes son heredados por la siguiente generación, es deseable que el conjunto reproductivo este compuesto por *buenos* individuos [Gol95].

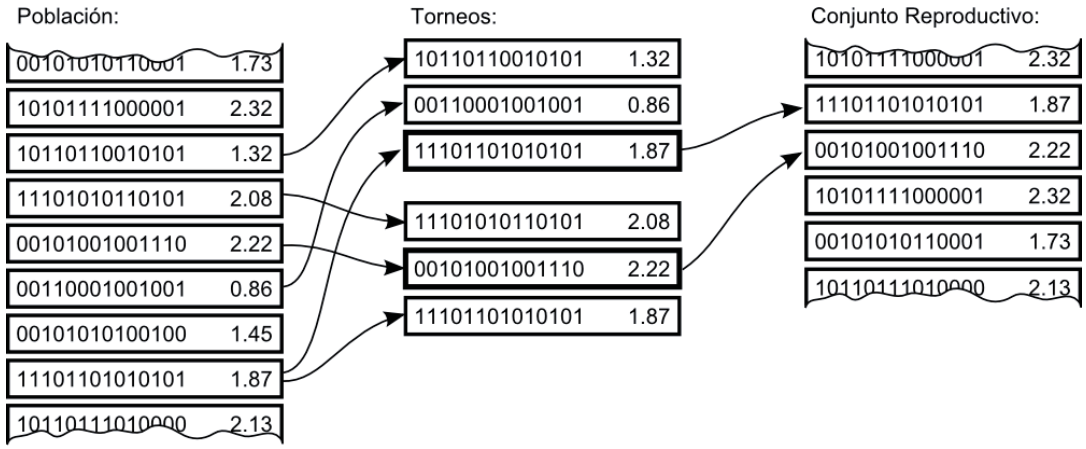


**Figura 2.7 – Selección por torneo.** *Fuente: Wikipedia (Autor: Lukipuk).*

La *presión de selección*, es el grado en que los mejores individuos son favorecidos: mientras más alta la presión de selección, más son favorecidos los mejores individuos. La tasa de convergencia está determinada en gran parte por la presión de selección, una alta presión de selección resulta en una alta tasa de convergencia. Los GA son capaces de identificar soluciones óptimas o cercanas a óptimas, bajo un rango de presión de selección amplio. Sin embargo, si la presión de selección es muy baja, la convergencia será lenta, por lo que el GA se tomara innecesariamente más tiempo para encontrar la solución óptima. Si la presión de selección es muy alta, es muy probable que el GA converja prematuramente a una solución sub-óptima.

La selección por torneo provee de una presión de selección al sostener un torneo entre *s* competidores, siendo *s* el tamaño del torneo. El ganador del torneo es el individuo con más aptitud de los *s* competidores, y luego el ganador se inserta en un conjunto reproductivo. El conjunto reproductivo, siendo compuesto de ganadores de torneo, tiene una aptitud promedio

más alta que la aptitud de la población. Esta diferencia de aptitud, provee la presión de selección, que impulsa al GA a mejorar la aptitud de cada generación. Para aumentar la presión de selección basta con aumentar el tamaño del torneo s, y como el ganador de un torneo más largo, en promedio, tendrá una aptitud más alta que un ganador de un torneo más pequeño.

#### 2.3.1.3Selección por truncamiento

Comparado a otros mecanismos anteriores que modelan la selección natural, la selección por truncamiento es un mecanismo de selección artificial. Este mecanismo es usado por criadores para la selección de individuos provenientes de poblaciones muy grandes. En este mecanismo de selección, los individuos son ordenados acorde a su aptitud. Solo los mejores individuos son seleccionados para el cruzamiento.

El parámetro para la selección por truncamiento es el *umbral de truncamiento*, el cual indica la proporción de población a ser seleccionada. Individuos por debajo del umbral de truncamiento no producen descendencia. La presión de selección disminuye con el aumento del umbral de truncamiento.

### 2.3.2 Mecanismos de cruzamiento

A causa de su importancia para el funcionamiento de los GA, una buena cantidad de literatura trata sobre distintas técnicas de cruzamiento su análisis. A continuación se discutirán algunas:

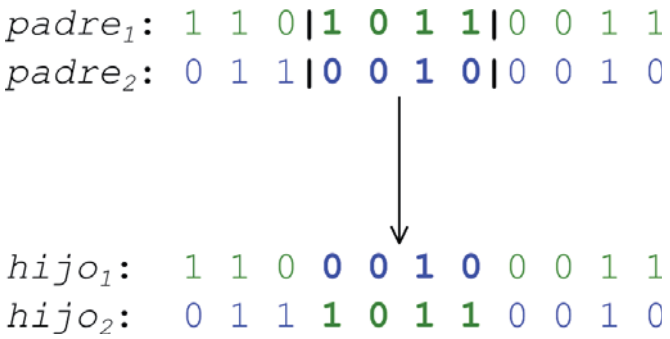


**Figura 2.8 – Cruzamiento de dos puntos.** *Fuente: Elaboración propia.*

Tradicionalmente, investigadores de GA definen el número de puntos de cruzamiento a uno o dos. En el cruzamiento de dos puntos, son elegidos dos puntos aleatoriamente, y el segmento comprendido entre estos puntos, es intercambiado (Figura 2.8).

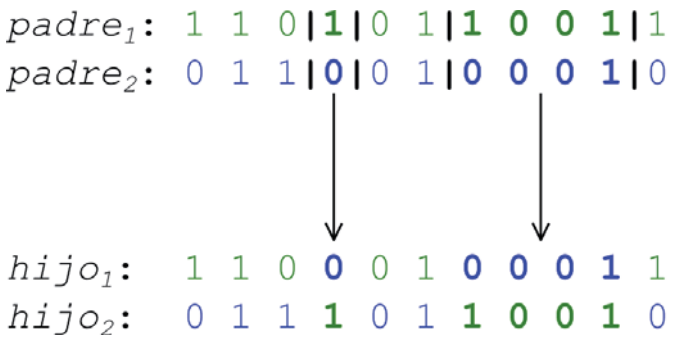


**Figura 2.9 – Cruzamiento multipunto.** *Fuente: Elaboración propia.*

Como extensión a este mecanismo de dos puntos, el cruzamiento multipunto, trata cada cadena o individuo como un anillo de bits dividido por *k* puntos de cruza, resultando *k* segmentos. Un conjunto de segmentos alternados se intercambia entre los individuos a cruzar (Figura 2.9).

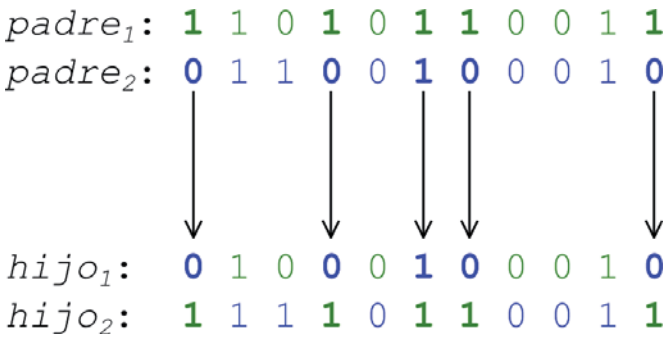


**Figura 2.10 – Cruzamiento uniforme.** *Fuente: Elaboración propia.*

El cruzamiento uniforme intercambia bits de una cadena, en vez de segmentos. En cada posición de la cadena, los bits son intercambiados con una probabilidad pre-establecida. E intercambio de bits en una posición, es independiente del intercambio en otra (Figura 2.10).

Estudios teóricos y experimentales han comparado distintas técnicas de cruzamiento, en particular, la cruza de uniforme y el de dos puntos. Por una parte el cruzamiento uniforme intercambia bits sin respetar su posición, pero su naturaleza disruptiva comúnmente se convierte en un inconveniente. El cruzamiento de uno y dos puntos preservan los esquemas, gracias a sus bajas tasas de alteración, pero luego se transforman en menos exploratorias cuando la población se torna más homogénea.

Un problema relacionado es la interrelación entre el tamaño de la población y el tipo de cruzamiento. Resultados experimentales sugieren que el cruzamiento uniforme es más adecuado para poblaciones pequeñas, mientras que la menos disruptiva cruza de dos puntos es más adecuada para poblaciones más grandes [Sri94]. La disruptividad del cruzamiento uniforme ayuda a sostener una búsqueda explorativa alta en pequeñas poblaciones. La

diversidad inherente de las poblaciones grandes reduce la necesidad de exploración, lo que hace al cruzamiento de dos puntos más adecuado.

# 3 Asignación de espacios en un patio de contenedores

Existen diversos aspectos que son deseables en los procesos operativos de un terminal portuario, específicamente en el patio de contenedores (el cual sirve de *buffer* para las operaciones de importación y exportación). La mayoría proviene de elementos directamente relacionados con el patio, pero no son propios de este. En este caso, se tomarán algunos aspectos de ambos, considerando asuntos exclusivamente de patios de contenedores y algunos aspectos que estén directamente relacionados con estos. Qué elementos se tomarán y cuáles no, serán descritos en el capítulo 4. En este capítulo se describen algunos procesos relacionados con el patio de contenedores, principalmente los de exportación e importación. Esto con el objetivo de contextualizar el problema y permitir una mayor comprensión del escenario formal a tomar en cuenta en la solución.

## 3.1 Conceptos básicos

Para contextualizar el ámbito del proyecto, a continuación se presentan los conceptos básicos de las operaciones de un terminal de contenedores.

### 3.1.1 Puerto marítimo

Físicamente, un puerto es una ubicación en la costa con la infraestructura necesaria para qué naves pueden atracar para la transferencia de personas y principalmente para el transporte de carga. Funcionalmente, son interfaces entre distintos modos de transporte. Son sistemas multifuncionales y hoy en día son verdaderos puntos de distribución de mercancías. Juegan un papel importante en el comercio, dado que gran parte del transporte de mercancías se hace a través del medio marítimo.

### 3.1.2 Puertos en Chile

Chile, por sus peculiares características geográficas, posee un borde costero muy extenso. En general la costa chilena se caracteriza por ser abierta, desabrigada y de poca profundidad, sin embargo se ha desarrollado la infraestructura necesaria para permitir el

transporte de todo tipo de carga, al construir un total de 56 puertos a lo largo del país. Cada zona del país se caracteriza por ciertos tipos de productos, así mismo los puertos en estas zonas sirven para estos mismos propósitos. Para la zona norte, los puertos son usados principalmente para el transporte de minerales dada la gran actividad de extracción de mineral. En la zona central se usan en general para productos agrícolas e importación de carga de valor agregado dado que se concentra la mayoría de la población del país. Más al sur están los puertos dedicados al transporte de productos derivados de la madera.

Los puertos públicos eran propiedad de la Empresa Portuaria de Chile (EMPORCHI), empresa estatal encargada de su administración, mantención y explotación. Mediante la Ley N° 19.542, publicada en diciembre del año 1997 [web2], se moderniza el sector portuario estatal, mediante la creación de 10 empresas portuarias del estado con patrimonio propio. El objetivo de estas empresas es la explotación, desarrollo y conservación de los puertos y terminales, así como de los bienes que posean incluidas las actividades inherentes al ámbito portuario. Por lo tanto estas empresas pueden efectuar cualquier tipo de estudio, proyecto, y ejecutar obras de construcción, ampliación, mejoramiento, conservación, reparación de puertos y terminales. Estas empresas son las siguientes, ordenadas geográficamente de norte a sur:

- Empresa Portuaria Arica.
- Empresa Portuaria Iquique.
- Empresa Portuaria Antofagasta.
- Empresa Portuaria Coquimbo.
- Empresa Portuaria Valparaíso.
- Empresa Portuaria San Antonio.
- Empresa Portuaria Talcahuano San Vicente.
- Empresa Portuaria Puerto Montt.
- Empresa Portuaria Chacabuco.
- Empresa Portuaria Austral.

Además de los puertos estatales existen 46 puertos privados, donde se distinguen entre puertos de uso privado y uso público. Los puertos privados de uso público trabajan de manera similar a los puertos estatales, pero son de propiedad privada. Estos explotan y administran su infraestructura bajo una concesión marítima. Los puertos privados de uso privado

corresponden a aquellos que prestan servicio a una empresa determinada y su existencia es consecuencia de las labores que esta desarrolla.

### 3.1.3 Equipo

A continuación se describen algunas de las máquinas necesarias para las actividades en un terminal de contenedores. Fundamentales para las faenas de carga y descarga, las grúas son máquinas que tienen como propósito levantar una carga, ya sea para cambiarla de posición dentro de un espacio, o dejarla en otra máquina para que esta la transporte. Otro equipamiento importante son los tracto camiones los cuales se utilizan para el tránsito de contenedores dentro del puerto como para el transporte hacia el exterior. Para el caso de las grúas, en un patio de contenedores se utilizan de distintos tipos según sus funciones, características, y capacidades.

Las grúas de muelle o *Quay Crane* se encuentran en el lado del muelle del terminal portuario y se encarga de la carga y descarga de contenedores de los barcos atracados en el muelle (Figura 3.1).


**Figura 3.1 – Quay Crane.** *Fuente: portstrategy.com*

Las grúas de patio tienen el propósito de ubicar y extraer los contenedores dentro del patio de contenedores. Existen de distintos tipos: las *Rubber Tired Gantry Crane* (Figura 3.3) y *Rail Mounted Gantry Crane* (Figura 3.2), ambas del tipo pórtico, difiriendo de su plataforma de movimiento, además están las *Reachstacker* y las *Toplifter*.


**Figura 3.2 – Rail Mounted Gantry Crane.** *Fuente: portstrategy.com*

Las *Rail Mounted* (Montada sobre rieles), como su nombre indica, se desplazan sobre rieles. Su movimiento está limitado a un eje que se extiende a lo largo del riel. Se utilizan específicamente para el manejo de contenedores de patio, utilizan energía eléctrica, por lo que su operación es más limpia. La desventaja es el costo de la infraestructura necesaria. Las *Rubber Tired*, en cambio, como su nombre lo indica, se movilizan a través de ruedas, son más lentas que las *Rail Mounted*, pero más contaminantes y ruidosas.

**Figura 3.3 – Rubber Tired Gantry Crane.** *Fuente: portstrategy.com*

Las *Reachstacker* son grúas pequeñas más frecuentes en puertos que tratan menores volúmenes de contenedores, son del tipo tractor. Estas son capaces de apilar hasta aproximadamente 3 contenedores. Las *Toplifter* son del mismo tipo con la diferencia de que se usan para apilar contenedores vacíos, por lo que pueden apilar hasta un aproximado de 7 contenedores. El puerto a considerar usa solo grúas RMG para el patio.

### 3.1.4 Patio de contenedores

En general el patio de contenedores de un puerto se encuentra dividido en yardas destinadas para almacenamiento de contenedores. Dentro de cada yarda se utiliza el mismo sistema que se usa en la estiba de barcos: el sistema BAROTI, el cual se explica en [web3]. Para el caso del patio de contenedores, este sistema de coordenadas de 3 ejes, como lo ilustra la figura 2.3, divide un sector en *bays*, los cuales corresponden a las filas transversales respecto al suelo y que se enumeran cada 20 pies; en *rows*, que corresponden a la ubicación del contenedor en un *bay*, en sentido longitudinal respecto al suelo; y en *tiers*, los cuales corresponden al nivel en el cual se encuentran los contenedores dentro de un sector.


**Figura 3.4 – Sistema BAROTI.** *Fuente: containerhandbuch.de*

## 3.2 Descripción de operaciones de patio

El modelo a resolver no tomará en cuenta ningún puerto en específico, sin embargo tomará algunos aspectos de los puertos en común. A continuación de describen los procesos relacionados con el patio de contenedores. Las descripciones de estas operaciones utilizadas para contextualizar el problema son extraídas de [Cota12], las cuales corresponden al terminal STI de San Antonio, Chile. Algunos aspectos son omitidos ya que no son útiles para la contextualización del problema.

### 3.2.1 Proceso de exportación

El proceso de exportación se inicia 72 horas antes del arribo del barco con la llegada de los contenedores al sector llamado "sector de p*re-stacking"*, el cual consiste en un sector en el patio de contenedores especializado para almacenar temporalmente los contenedores de exportación traídos por los clientes. En este sector, los contenedores son ordenados de manera inversa a la estiba del barco, de manera de evitar movimientos innecesarios.

Aquí la descarga de estos camiones es realizada por una grúa *reachstacker* (descritos en este capítulo). El plan de estiba se diseña considerando la estabilidad del barco en el proceso de carga, para esto debe considerarse el peso de los contenedores y el tipo de éstos, para lograr un centro de masa los más equilibrado posible, para no comprometer la integridad del proceso de transporte. Los mismos aspectos deben considerarse para mantener la integridad los mismos contenedores y/o su contenido. De este mismo modo, el puerto de destino de los contenedores también es un aspecto importante al momento de planificar la estiba. Se debe planificar tratando de satisfacer todos estos aspectos.

Cuando el barco atraca en el puerto, los contenedores van siendo cargados en los tracto camiones por las *reachstacker*, para ser llevados al sector del muelle donde las grúas de muelle (QCs) toman el contenedor y lo embarcan en el barco. Es importante señalar que STI mantiene una política de atención a sus clientes, la cual consiste en que los contenedores son clasificados mediante segregaciones en donde el barco, puerto de destino, peso, largo (20 o 40 pies) y tipo (*reefer*, carga peligrosa) determinan su segregación.

### 3.2.2 Proceso de importación

Con el arribo de un barco al puerto se da inicio al proceso de importación. Cuando el barco ha atracado se inicia la descarga de los contenedores que deben ser descargados en ese puerto. Esta actividad se lleva a cabo por las *QCs*. A medida que son descargados del barco, los contenedores son cargados en los tracto camiones internos, los cuales deben transportar los contenedores desde el muelle hasta el sector del patio asignado con anterioridad. Una vez en el sector pre-asignado de importación, el camión es descargado por una grúa *reachstacker* (sector importación, carga directa) o *RTG* (sector importación carga indirecta) para ser dejado en uno de los *stacks*, en donde se puede tener contenedores de carga directa, indirecta o vacíos. El tracto camión, una vez descargado está libre para ir al sitio del barco para ser cargado nuevamente.

Se entiende por carga directa a la carga que es retirada por los clientes a lo más 24 horas después del arribo del barco. En este caso, el retiro del contenedor es coordinado entre el terminal y el cliente mediante un sistema de citas, donde se establece una hora en la que el cliente puede retirar el contenedor. Si el contenedor, de carga directa, sobrepasara 48 horas en

el patio del puerto, se le aplica una tarifa. La carga indirecta es definida como el tipo de carga que no está programada para ser retirada por los clientes antes de las primeras 24 horas. El motivo es de carácter económico, y no viene al caso. Un contenedor, por ley, puede permanecer en el patio hasta un máximo de 90 días. El retiro de este tipo de carga sigue el mismo sistema de citas que la carga directa. Los contenedores vacíos que llegan al puerto son almacenados, hasta que los propietarios los retiren a los depósitos de contenedores. Lo que motiva la entrada y salida de contenedores vacíos, es que existe una diferencia entre la importación y exportación, por lo mismo el ingreso de los contenedores vacíos tiene como objetivo ser usados para cargarse para exportación.

## 3.3 Trabajos relacionados

Antes de revisar el modelo formal del problema, en esta sección se mostrará el resultado de la revisión de literatura efectuada en el marco del proyecto de título. Se hace un resumen de las partes más relevantes de las publicaciones las cuales son: modelo, solución propuesta, resultados y conclusiones. Algunos trabajos están aquí a causa de su relación con el ámbito del problema a resolver y proponer soluciones en base a GA. A pesar que algunos proponen soluciones y/o métodos ajenos a los GA, estos sirven para contextualizar el problema y además para comparar estos métodos de solución con los GA. Se explica de forma más extensa y detallada aquellos trabajos que utilicen algoritmos genéticos.

### 3.3.1 Planificación y gestión de operaciones portuarias en el patio de contenedores

En la memoria final para optar al título de ingeniero civil industrial de Julián Elorrieta [Elo11], se estudian las operaciones del puerto de San Antonio, específicamente las de la empresa STI. El objetivo es determinar los espacios donde deben están ubicados los contenedores, para aumentar la productividad de las operaciones de carga y descarga. El factor que determina la productividad es el tiempo que permanece una nave en el puerto.

Se propone un modelo de dos niveles, de acuerdo en dos factores que permiten aumentar la productividad, el primero es la cercanía entre los contenedores que serán embarcados en la misma nave, y el segundo es el orden de las operaciones de carga y descarga, de tal manera de evitar la interferencia entre las grúas. Para la resolución se propone un esquema de horizonte rodante, cuya ventaja es la posibilidad de insertar el modelo en la operación del puerto.

El modelo fue resuelto con CPLEX 9.0. Las pruebas iniciales no fueron satisfactorias, ya que el modelo es muy extenso y produjo problemas de falta de memoria. El modelo fue modificado al usar unidades más grandes de espacio, permitiendo la resolución en un tiempo razonable (1 hora y 18 minutos).

### 3.3.2 A Genetic Algorithm to Solve the Storage Space Allocation Problem in a Container Terminal

Esta publicación hecha por *Mohammad Bazzazi*, *Nima Safaei* y *Nikbakhsh Javadian* [Bsj08], presenta un algoritmo genético para resolver el SSAP. Este problema se define como la asignación temporal de contenedores de entrada y salida a los bloques de almacenamiento en cada período de tiempo, con objetivo de balancear la carga entre bloques para minimizar los tiempos de almacenaje y retiro de los contenedores. En este trabajo se considera una versión extendida del SSAP, en donde el tipo de contenedor afecta en la decisión de asignar los contenedores a los bloques. En casos reales, existen distintos tipos y tamaños de contenedores. El SSAP extendido es resuelto por un algoritmo genético eficiente para instancias de la vida real. En este trabajo se consideran dos tipos de contenedores:

- Contenedores de importación que se encuentran en barcos antes de ser descargados y asignados al patio de contenedores (C1).
- Contenedores que ya están en el patio de contenedores a la espera de ser retirados por los clientes (C2).

El objetivo de este trabajo es minimizar el tiempo promedio en que un barco permanece en el terminal, lo que es una medida del servicio de un terminal a las navieras.

En el terminal de contenedores considerado en este trabajo, el 85% del total de la carga de trabajo está relacionado a los contenedores C1 y C2 (Contenedores de importación). Los de exportación son ubicados en un solo bloque cercano al sitio del barco donde estos serán embarcados. Esa es la razón por lo cual no se consideran contenedores de exportación. Aquí se usa un enfoque de horizonte rodante, donde el horizonte de planificación es de cuatro días. Cada día está dividido en 6 períodos de 4 horas. Al comienzo del día 1, un plan de asignación de espacio se determina para los 24 períodos en los días 1-5. Solo el primer día del plan es ejecutado y un nuevo plan de cuatro días se determina al fin del primer día, basado en la información más actualizada. Esto se repite para cada día.

Existen contenedores con tiempos de despacho desconocidos al tiempo de planificación o contenedores con tiempos de despacho que están fuera del horizonte de planificación. Sus cargas de trabajos no se hacen efectivas dentro del horizonte de planificación, en consecuencia aquellos contenedores no son directamente incluidos en el

modelo de asignación de espacios. Para tener en cuenta su posible efecto en el futuro, estos contenedores son distribuidos en bloques en proporción del espacio disponible de cada bloque al principio del horizonte de planificación. Esta aproximación tiene un efecto marginal en el rendimiento general. La mayoría de los contenedores son asignados bajo información conocida. Bajo esta descripción se requieren cuatros tipos de datos para cada contenedor de importación: el tiempo en que el contenedor debe ser descargado del barco y llevado a los bloques de almacenamiento, el tipo del contenedor y los bloques disponibles para ese contenedor. En el trabajo afirman que el balance de cargas en los bloques, asegura la disminución de los tiempos de permanencia.

### 3.3.2.1Definición formal

El SSAP extendido se formula como un modelo de programación matemática basado en las siguientes suposiciones. Básicamente, se debe determinar el número de contenedores C1 y C2 almacenados en cada bloque para cada período de planificación.

- Para la carga de trabajo solo se consideran contenedores de importación (C1 y C2), los que deben ser reubicados del barco al patio de contenedores.
- Existen suficientes recursos (Grúas de patio, grúas de muelle y camiones) para manejar las cargas de trabajos en el terminal considerado.
- Los contenedores son de distintos tipos y tamaños. Las cargas de trabajos para cada tipo y tamaño son medidos en términos del número de contenedores de ese tipo. Los contenedores de distinto tipo y tamaño pueden ser mezclados dentro de un bloque.
- Los bloques disponibles a cada tipo se conocen de antemano.

**Parámetros de entrada**

$B$ El número total de bloques en el patio de contenedores

$T$ El número de período s de planificación dentro del horizonte de planificación, $T=24$

$R$ La cantidad de tipos de contenedores

$C_i$ La capacidad del bloque $i$, $1 \leq i \leq B$

$\tilde{D}_{tkr}$ La cantidad esperada de contenedores C1 de tipo $r$ que son descargados en el período $t$ y son retirados por clientes en el período $t + k$, $1 \leq t \leq T$, $0 \leq k \leq T - t$

$\beta_{itr}$ La cantidad esperada de contenedores C1 de tipo *r* descargados en el período *t*, asignados al bloque *i*, con tiempos de retiro desconocido o con tiempos de retiro fuera del horizonte de planificación, $1 \leq t \leq T,\ 1 \leq i \leq B$

$P^{0}_{itr}$ La cantidad total esperada de contenedores C2 de tipo *r* que llegan al terminal en el período *t* que deben ser embarcados en el período $t + k,\ 1 \leq t \leq T,\ 0 \leq k \leq T - t$

$V_{ir}$ El inventario inicial del tipo *r* en el bloque *i*, $1 \leq i \leq B$

$S_{ir} = 1$ Si el contenedor tipo *r* puede ser asignado al bloque *i*; de otra manera $S_{ir} = 0$

$\eta$ Densidad permitida para cada bloque

$M$ Un entero positivo arbitrariamente grande

**Variables de decisión**

$D_{itkr}$ La cantidad de contenedores C1 de tipo *r* con información completa almacenados en el bloque *i* que son descargados de los barcos en el período *t* y que deben ser retirados en el período $t + k,\ 1 \leq i \leq B,\ 1 \leq t \leq T,\ 0 \leq k \leq T - t$

$D_{itr}$ El número total de contenedores C1 de tipo *r* (de información completa o parcial) almacenados en el bloque *i* que son descargados en el período $t,\ 1 \leq i \leq B,\ 1 \leq t \leq T$

$P_{itr}$ El número total de contenedores C2 de tipo *r* almacenados en el bloque *i* que son retirados en el período $t,\ 1 \leq i \leq B,\ 1 \leq t \leq T$

$V_{itr}$ El número de contenedores del tipo *r* (C1 y C2) dentro del bloque *i* al final del período $t,\ 1 \leq i \leq B,\ 1 \leq t \leq T$

**Modelo matemático**

La función objetivo se denota como:

$$\min Z = \sum_{t=1}^{T}\sum_{r=i}^{R}\left(w_1\left[\max_{1\leq i\leq B}\{D_{itr}\} - \min_{1\leq i\leq B}\{D_{itr}\}\right] + w_2\left[\max_{1\leq i\leq B}\{D_{itr} - P_{itr}\} - \min_{1\leq i\leq B}\{D_{itr} - P_{itr}\}\right]\right)$$

La función objetivo busca el balance de los contenedores C1 y el total de contenedores entre los bloques para cada tipo de contenedor y cada período de tiempo. El término *"max(.) – min(.)"* implica el desbalance entre las cargas de trabajo de cada bloque. Por lo tanto, se busca que *"max(.) = min(.)"*, y quiere decir que los contenedores están distribuidos uniformemente entre los bloques. El primer término se enfoca en el balance de los contenedores C1, y el segundo término se enfoca en el balance de los contenedores C1 y C2 simultáneamente. La importancia de cada termino está dada por los coeficientes de peso $w_1$ y

$w_2$. En la ecuación, $D_{itr}$ es el total de contenedores relacionados a los barcos que deben ser almacenados en el bloque *i* en el período *t* y $D_{itr} + P_{itr}$ es el número total de contenedores (relacionados a los barcos y a los clientes) a ser almacenados en el bloque *i* en el período *t*. Por lo tanto, los dos términos en la ecuación miden el desbalance de los contenedores de descarga y el desbalance del total de contenedores en cada período de planificación, respectivamente. Los pesos de los términos, $w_1$ y $w_2$ se ajustan dependiendo de la importancia relativa, según como lo interprete el terminal. Sujeto a:

(1)

$$\sum_{i=1}^{B} D_{itkr} = \tilde{D}_{tkr}$$

Esta restricción (1) asegura que el total de contenedores C1 de tipo *r* de información completa esperando a ser asignado, $D_{tkr}$, es la suma de los contenedores asignados a todos los bloques.

(2)

$$D_{itr} = \beta_{itr} + \sum_{k=0}^{T-t} D_{itkr}$$

Esta restricción (2) asegura que el total de contenedores C1 de tipo *r* asignados al bloque *i* durante el período *t*, $D_{itr}$, es la suma del total de contenedores C1 de información completa, $D_{itkr}$, y aquellos contenedores con tiempo de retiro desconocido en el horizonte de planificación, $\beta_{itr}$.

(3)

$$P_{itr} = \sum_{k=0}^{t-1} D_{i(t-k)kr} + P_{itr}^{0}$$

Esta restricción (3) indica que el número de contenedores C2 de tipo *r* asignados en el bloque *i* durante el período *t*, $P_{itr}$, consiste de dos partes. La primera son los contenedores transferidos desde los contenedores C2 correspondientes que llegan en el horizonte de planificación. La segunda parte consiste en contenedores C2 inicialmente almacenados en el bloque *i* para ser cargado en barcos en el período *t* en el horizonte de planificación actual.

(4)

$$V_{itr} = V_{i(t-1)r} + \tilde{D}_{itr} - P_{itr}$$

(5)

$$V_{itr} \leq \eta C_i$$

La restricción (4) representa la actualización de inventario, $V_{itr}$, período a período. La restricción (5) asegura que el inventario de cada bloque en cada período no exceda la densidad permitida. La densidad del bloque implica el hecho que una porción del bloque debe ser usado para transferir contenedores dentro del bloque.

(6)

$$\tilde{D}_{itr} \leq MS_{ir}$$

La restricción (6) asegura que cada tipo de contenedor poder ser asignado solo a los bloques permitidos.

### 3.3.2.2Algoritmo genético

Se escoge esta técnica dada su eficiencia para este tipo de problemas y el SSAP no había sido resuelto mediante esta técnica. Considerando la principal variable de decisión, $D_{itkr}$, se usa una estructura de 4 dimensiones para representar la solución del SSAP extendido. Estas 4 dimensiones indican los índices relacionados al bloque, período de descarga, período de despacho y tipo de contenedor.

**Generación de la población inicial**

Para generar soluciones aleatorias factibles para la población inicial, proponen el siguiente procedimiento:

- Generar valores $D_{itkr}$ dentro del intervalo $[max\{D_{itkr}\}, min\{D_{itkr}\}]$ donde $\sum_{i=1}^{B} S_{ir} D_{itkr} = \tilde{D}_{tkr} \ \forall i,t,k,r$. Esto se hace a través de combinaciones lineales.
- Calcular valores $D_{itr}$ usando la restricción (2).
- Calcular valores $P_{itr}$ usando la restricción (3).
- Calcular valores $V_{itr}$ usando la restricción (4). Considerando la restricción (5). Si la restricción (5) es violada, entonces se agrega un valor $\lambda(V_{itr} - \eta C_i)$ a la función objetivo como una penalización donde $\lambda$ indica el coeficiente de penalización.

**Cruzamiento aritmético**

Para mantener la factibilidad de las nuevas generaciones se propone el uso del siguiente mecanismo como operador de cruza. La cruza aritmética explora el espacio de solución manteniendo la factibilidad simultáneamente. Este operador genera la descendencia como la combinación lineal complementaria de los padres de la siguiente forma:

Descendencia ≡ $\lambda$ x $Padre_1$ + (1 – $\lambda$) x $Padre_2$

Donde $\lambda$ es un valor aleatorio dentro del intervalo [0, 1]. Así, los valores $D_{itkr}$ son generados como:

$D_{itkr}$(descendencia) = $\lambda$ x $D_{itkr}(Padre_1)$ + (1 – $\lambda$) x $D_{itkr}(Padre_2)$

Este mecanismo garantiza que la descendencia se mantendrá factible si sus padres lo son. Ocurre un problema ya que estos valores deben ser enteros. Por lo tanto la restricción (1) podría no cumplirse y se incurre en un error . Para resolver esto, el valor de ε se suma heurísticamente al $D_{itkr}$ con el valor mínimo.

$$\varepsilon = \sum_{i=1}^{B} [D_{itkr}(des)] - \tilde{D}_{tkr}$$

**Mutación**

La principal tarea de la mutación es mantener la diversidad de la población al pasar las generaciones. En este trabajo se utiliza el mecanismo de mutación llamado *"Stepping Stone mutation"*. Este mecanismo asegura que las soluciones mutadas se mantendrán factibles si la solución sujeta a mutación lo es. El procedimiento es el siguiente:

- Seleccionar aleatoriamente una solución.
- Cambiar los valores de $D_{ptkr}$ y $D_{qtkr}$ a $D_{ptkr} \rightarrow D_{ptkr} - \left[\frac{\delta_{tkr}}{2}\right]$ y $D_{\downarrow}qtkr \rightarrow D_{\downarrow}qtkr - \{\delta_{\downarrow}tkr - [\delta_{\downarrow}tkr/2]\}$, respectivamente.

**Selección**

Para la selección, se utiliza el típico mecanismo de selección por ruleta, explicada en detalle en el capítulo 3.

**Estrategia de aceptación de descendencia**

Se usa una estrategia tipo *semi-greedy* para aceptar las descendencias. En este caso una descendencia es aceptada para la siguiente generación si su aptitud es menor que la del promedio de sus padres.

**Criterio de término**

Aquí se usan dos criterios: el típico mediante ciertas cantidad de generaciones y una cierta desviación estándar de los valores de aptitud de los genotipos de la generación actual.

#### 3.3.2.3Resultados

Se efectuaron pruebas para 22 set de datos, con distintos tamaños, y para comparar se utilizó el método de Branch & Bound. Este método solo pudo resolver los set de datos más pequeños, ya que para los set más grandes se excedió en el tiempo razonable (3 horas). El algoritmo genético no tuvo problemas en resolver todo el set. El set más grande que pudo resolver el método B&B (7.650 segundos), fue resuelto por el GA en 30 segundos. La implementación del GA para este problema tuvo excelentes resultados.

### 3.3.3 Using Genetic Algorithms to Solve the Yard Allocation Problem

Ping Chen, Zhaohui Fu y Andrew Lim, del departamento de ciencias de la computación de la Universidad de Singapur, a través de dos publicaciones, presentan el *Yard Allocation Problem*. El primero estudia el uso de distintas heurísticas (Tabu search, Squeaky Wheel y Simulated Annealing) para resolver el problema formulado [Cfl02], el segundo toma la misma formulación que la publicación anterior, pero esta vez aplicando un algoritmo genético [Cfl02-1]. El YAP tiene una formulación matemática simple, no por ello una solución trivial. Su dificultad está en la evaluación de la función objetivo.

El objetivo del Yard Allocation Problem puede ser expresado de dos maneras:

1. Minimizar el espacio de patio usado para acomodar todos los requisitos de espacio.
2. Maximizar el número de requerimientos asignados en un espacio fijo.

Estos objetivos son de hecho similares el uno al otro. Se usará el primero como objetivo en este caso ya que es de interés, el satisfacer todas las solicitudes. El cómo se gestiona las solicitudes y como se actúa en caso de rechazar una solicitud, están fuera del alcance del problema.

Sea $R$, un conjunto de $n$ requerimientos de espacio y $E$, un patio de contenedores de espacio infinito. $\forall R_i \in R$, $R_i$ tiene una serie de requisitos de espacio $Y_{ij}$, de largo $L_{ij}$, $j \in [T_{i\ inicio}, T_{i\ fin}]$. Considere una función de mapeo $F$, tal que $F(Y_{ij}) = e_k$, donde $e_k \in E$ es una posición en $E$.

La función a minimizar es la siguiente:

$$\max_{\forall R_i \in R,\ \forall Y_{ij} \in R_i} \left(F\left(Y_{ij}\right) + L_{ij}\right)$$

Sujeto a:

a) $p = q - 1$; $\quad (\forall p, q \in [T_{i\ inicio}, T_{i\ fin}])$.

b) $F(Y_{ip}) \geq F(Y_{iq})$;

c) $F(Y_{ip}) + L_{ip} \leq F(Y_{iq}) + L_{iq}$;

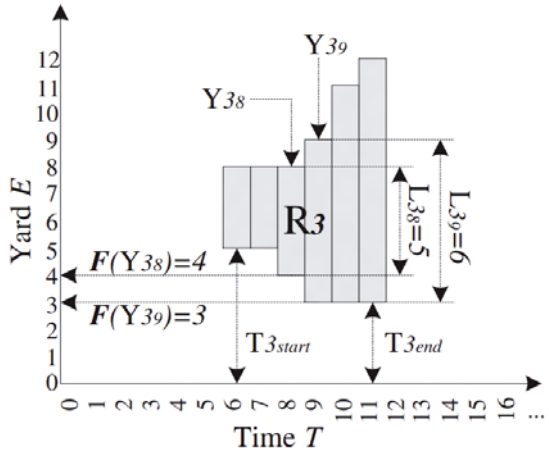


**Figura 3.5 – Una solicitud representada por un SLS.** *Fuente: [Cfl02]*

La figura 3.5 ilustra el ejemplo que usaremos. Aquí se muestra una distribución con una solicitud *válida* $R_3$. El patio $E$ es tratado como una línea infinita. El tiempo $T$ se convierte en una variable discreta, con 1 como unidad mínima. $R_3$ posee seis requerimientos de espacio en el intervalo [6, 11] ($T_{3\ inicio} = 6$, $T_{3\ fin} = 11$). La posición para $Y_{38}$ y $Y_{39}$ son $F(Y_{38}) = 4$ and

$F(Y_{39}) = 3$ respectivamente. La salida de la función de mapeo para $R_3$ ese (5, 5, 4, 3, 3, 3). Se ve que todas las restricciones se cumplen: $F(Y) \geq F(Y)$ y $F(Y) + L \leq L$. El *máximo* viene de $Y_{3\text{-}11}$ viene de $Y_{3\text{-}11}$, donde $F(Y_{3\text{-}11}) + L_{3\text{-}11} = 12$.

Las solicitudes serán llamadas SLS (figuras con forma de escalera, del inglés *Stair-Like Shapes*) debido al parecido que forman las solicitudes al proyectarlas en el eje del tiempo, en un plano cartesiano, con una escalera [Cfl02].

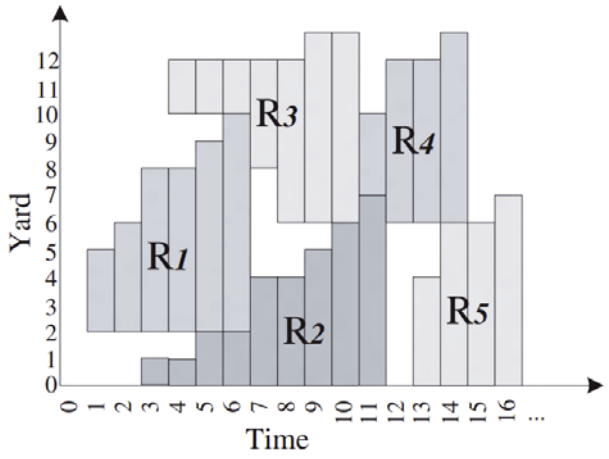


**Figura 3.6 – Cinco solicitudes en una representación válida.** *Fuente: [Cfl02]*

La figura 3.6 ilustra el problema de forma geométrica. Sin embargo, esta representación podría resultar complicada de manipular por un algoritmo de solución. Por lo tanto, se transformará la configuración geométrica en un grafo. La figura 3.7 muestra el grafo resultante de esta transformación sobre la configuración de la figura 3.6.

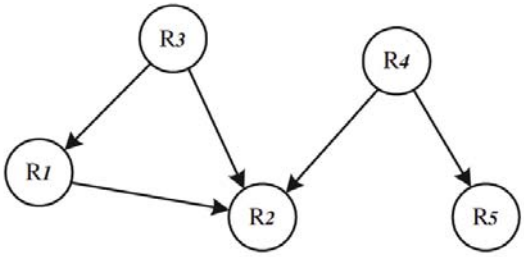


**Figura 3.7 – Configuración de solicitudes representadas en un grafo.** *Fuente: [Cfl02]*

Cada solicitud $R_i$ se representa como un vértice. Si en algún intervalo de tiempo, existe solapamiento entre dos SLSs, se representa con una arista $E_{ij}$, la que conecta la solicitud $R_i$ con

la solicitud $R_j$. La dirección de la arista determina la posición relativa de las dos solicitudes en el patio físico. En la figura 3.6 se observa que, $R_1$ y $R_2$ requieren una cantidad de espacio, en al tiempo 3, 4, 5 y 6, en la figura 3.7 se ve que hay una arista entre $R_1$ y $R_2$. Ya que $R_1$ está ubicada por sobre $R_2$, la dirección de la arista va desde $R_1$ a $R_2$, a esta arista se le llama $E_{12}$. Claramente, el grafo resultante es un DAG (Grafo dirigido acíclico, del inglés *Directed Acyclic Graph*). En este DAG, a cada $R_i$ se le puede asignar un AL (Etiqueta acíclica, del inglés, Acyclic Label) $L_i$ tal que, una arista $E_{ij}$ implica que $AL(R_i) < AL(R_j)$. Tenga en cuenta que cada $AL(R_i)$ ($1 < i \leq n$) tiene valor único. De esta asignación de AL, surge un problema.

Cada configuración física podría tener una relación de correspondencia con más de una asignación de AL. Esta relación 1...N entre la configuración física y la asignación de AL traería grandes problemas de confusión en búsquedas heurísticas, incluyendo algoritmos genéticos. Los métodos heurísticos tienden a identificar ciertos *buenos* patrones que los podrían guiar a mejores soluciones, mientras exploran el espacio de búsqueda. Sin embargo al intentar normalizar estas relaciones, podría incluso empeorar el rendimiento de una heurística. Esto queda en evidencia en la publicación *Using Genetic Algorithms To Solve The Yard Allocation Problem* [Cfl02-1], donde los autores exponen la nula ventaja que resulta la reparación de este problema.

Volviendo al problema, la solución óptima no es más que la configuración física que resulta en el menor espacio máximo ocupado en todo el intervalo de tiempo evaluado. Naturalmente, esta configuración física óptima tiene una asignación de AL óptima. Por lo que el objetivo de la solución es encontrar esa asignación de AL óptima. Con esto surge otro problema, la evaluación de una solución o asignación de AL, la cual resulta una operación *no trivial*. En el SBP [Lim98], se usa el algoritmo del camino más largo en un DAG, para encontrar la longitud mínima de muelle requerido. Sin embargo el Yard Allocation Problem trata con estas formas irregulares, las SLS, las que su posición relativa y distancia no pueden ser calculadas de una forma directa y sencilla, como si se puede hacer con los rectángulos del caso del SBP. Chen, Fu y Lim [Cfl02], proponen un procedimiento recursivo para encontrar el espacio mínimo necesario para una asignación AL dada:

```
Procedimiento Evaluar solución(A)
1     Mientras exista un SLS sin asignar
2           Escoger el SLS "S" con él AL más grande
3           Soltar(S, S_fin, 0)
4     Para cada tiempo T_i hacer
5           Si T_i > L entonces
6                 L := T_i
7     Retornar L
Procedimiento Soltar(S, t, l)
1     L := Posición más baja para soltar los escalones(tiempo t')
2     Si L < l entonces
3           L := l
```

```
4     Para cada escalón s después de t' − 1 hacer
5           Soltar s a la posición L
6     Soltar(S, t' − 1, L)
```

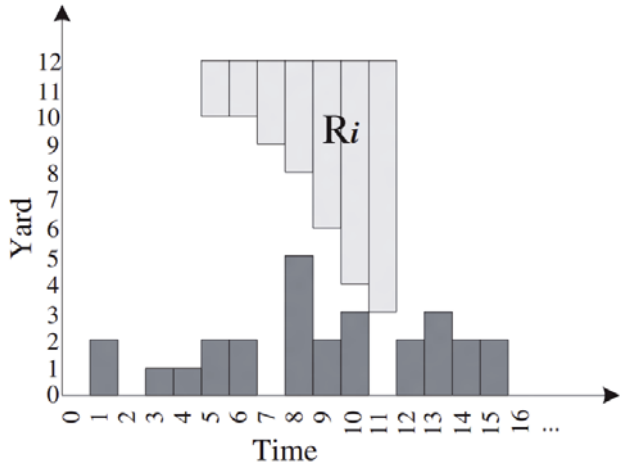


**Figura 3.8 – Una solicitud $R_i$ alineada al techo, antes de comenzar el proceso.** *Fuente: [Cfl02]*

El procedimiento recursivo *Soltar* usa un enfoque de tipo *greedy* para soltar un *escalón* (Un trozo de SLS en una unidad de tiempo) dado, a la posición más baja posible. Los detalles se ven en las figuras 3.5, 3.6 y 3.7. Primero $R_i$ es alineado al "cielo" antes que comience el proceso (Figura 3.5). Luego, del tiempo 5 al 11, encontramos la distancia máxima en que cada escalón puede caer, sin exceder un el límite inferior de 0. Se usa, la mínima de las distancias posibles en que un escalón puede caer. En este caso, la distancia mínima de 1 se da en el tiempo 10, por lo que cada escalón se deja caer en 1 (Figura 3.6).

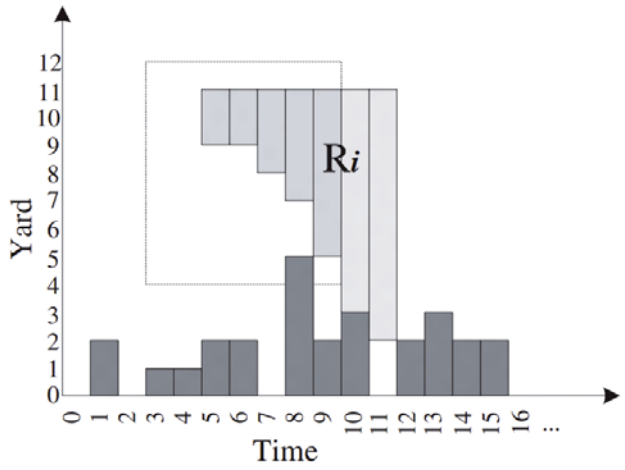


**Figura 3.9 – Cada escalón cae en 1. Los escalones al tiempo 10 y11 están en sus posiciones finales. Los escalones que pueden seguir cayendo, están destacadas con un color más oscuro, encerradas en un rectángulo.** *Fuente: [Cfl02]*

A causa del alineamiento hacia el techo inicial, no se requiere de más caídas desde el escalón 10 en adelante. Note que la *superficie* fue tocada al tiempo 10.

Los escalones de los tiempos 5 al 9, que están destacados en la figura 3.6, pueden seguir cayendo, pero esta vez, con un límite inferior de 3, el cual es la altura de la superficie impactada la tiempo 10. El proceso de caída se completa después de algunas recursiones en los tiempos 8, 7, 6, y 5. La configuración final se ve en la figura 3.7. En el peor escenario, la complejidad de tiempo para el procedimiento *Soltar* es de *n x T*, donde *n* es el número de solicitudes y T es el tiempo promedio de vida de todas las solicitudes.

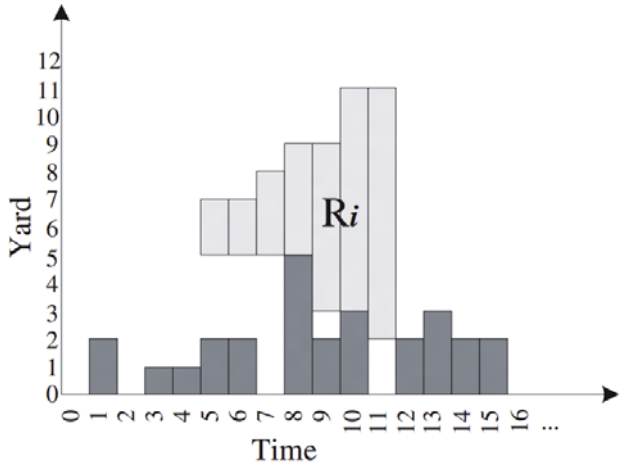


**Figura 3.10 – Cada escalón está en su posición final.** *Fuente: [Cfl02]*

Ahora con la evaluación para una asignación de AL dada, falta aplicar algún método para encontrar una asignación de AL óptima. En este caso, se diseñará y aplicará un algoritmo genético que explore el gigantesco espacio de búsqueda para encontrar la solución óptima(o una muy buena solución, en el peor de los casos) en un buen tiempo.

### 3.3.3.1Algoritmo Genético

Dada la naturaleza del problema a resolver por el algoritmo genético, es claro que la representación binaria clásica no es adecuada en YAP, donde la asignación de AL es usada como representación de la solución. El espacio de búsqueda es una permutación sin repetición. Los códigos binarios de estos AL (números enteros) no dan ninguna ventaja, incluso podría ser peor: el cambio de un solo bit le quitaría la factibilidad a la solución.

En este trabajo solo se presentan el genotipo y los mecanismos de cruza y mutación.

**Codificación**

La codificación del genotipo consiste simplemente en una cadena de números enteros desde el *0* hasta *n-1*, donde *n* es el número de solicitudes. Cada posición representa una solicitud $R_i$, y su valor numérico representa su AL en la representación en el grafo. Gráficamente se puede interpretar como el orden en que una la solicitud $R_i$ es removida. O viéndolo de forma inversa, el orden en que las SLS (formas de las solicitudes) son apiladas. Por ejemplo, un genotipo representando 7 solicitudes:

$g = R_1 R_2 R_3 R_4 R_5 R_6 R_7$

Una solución posible:

$g = 5\ 1\ 4\ 2\ 6\ 3\ 0$

Esto significa que el orden en que pueden ser removidas las solicitudes es:

$R_7 - R_2 - R_4 - R_6 - R_3 - R_1 - R_5$

**Partially Mapped Crossover (PMX)**

Esta técnica de cruzamiento fue propuesta para resolver el problema del vendedor viajero (TPS). Se hacen algunos ajustes para acomodar el genotipo. El PMX genera una descendencia tomando una sub secuencia de una asignación de AL proveniente de un padre y preservando el orden y posición de tantas AL como sea posible del otro padre. La sub secuencia se determina al escoger dos puntos de corte aleatorios. Por ejemplo, considere estos padres:

$p_1$ = 012|3456|789

$p_2$ = 312|5740|968

Estos generan las siguientes descendencias. Primero, los segmentos entre los puntos de corte se intercambian:

$o_1$ = ---|5740|---

$o_2$ = ---|3456|---

Este intercambio define una serie de mapeos:

3↔5, 4↔7, 5↔4, 6↔0

El resto se rellena con los AL originales de los padres, los que no tienen conflicto:

$o_1$ = -12|5740|-89

$o_2$ = -12|3456|9-8

Finalmente, la primera posición en $o_1$ (debería ser 0, pero causa conflicto, ya que el 0 ya se encuentra presente en la posición 7) es remplazada por el 6 a causa del mapeo 6↔0. Estos remplazos son transitivos hasta que el conflicto se resuelva. Por ejemplo, la segunda posición de $o_1$ debería seguir los mapeos 7↔4, 4↔5 y 5↔3, por lo que se remplaza por el 3. Las descendencias finales son:

$o_1$ = 612|5740|389

$o_2$ = 712|3456|908

**Mutación**

En este trabajo se habla de 4 métodos de mutación, sin embargo, no especifican cual se usa en las pruebas. La inversión, donde se invierte una sub secuencia. Inserción, donde se selecciona un AL y se inserta en otra posición aleatoria. Desplazamiento, donde se toma una sub secuencia y se inserta en otra posición aleatoria. Intercambio recíproco, donde se intercambian dos AL aleatorios.

**Otras especificaciones y resultados**

Se estableció el tamaño de la población en $P = 1000$ en la mayoría de los casos. La población inicial se genera aleatoriamente. La población se ordena de acuerdo a la función objetivo, mientras mejor es la solución, mayor es la probabilidad de ser seleccionada. En cada iteración, se produce una nueva generación de tamaño $2P$ donde la mejor mitad sobrevive a la siguiente iteración. El criterio de término no se especifica.

Los resultados indican que de todas las heurísticas sometidas a prueba, el GA es la aproximación más rápida. Cabe destacar que la rutina de normalización de las asignaciones de AL (*Acyclic Label*, ver sección 3.2), no mejoraron los resultados como creyeron que debería. En teoría la normalización debería hacer el proceso de búsqueda más estable al remover factores confusos. Pero en la práctica resultó que los efectos secundarios tienden a opacar sus beneficios.

### 3.3.4 The General Yard Allocation Problem

En esta ocasión, a Chen, Fu y Lim y Rodrigues, formulan un problema similar al YAP pero añadiéndole una dimensión adicional de espacio, ahora el problema considera un espacio tridimensional (dos de espacio, y una de tiempo) [Cflr03]. En cada ranura de tiempo se enfrenta un problema del tipo 2-DRPP (*Two Dimensional Rectangular Packing Problem*) con ciertas restricciones adicionales. Se sabe que el 2-DRPP es de clase NP-hard.

Se ponen a prueba distintas heurísticas, como *Tabu Search* (*Long* y *Short Term Memory*), *Squeaky Wheel Optimization*, *Simulated Annealing* y Algoritmos Genéticos. Nuevamente descubren que la implementación de un GA alcanza los mejores resultados para este problema.

# 4 Propuesta de solución

Este proyecto tendrá dos objetivos, determinar la ubicación de cada contenedor (bloque) con el fin de minimizar el tiempo de permanencia de una nave y planificar las ubicaciones de contenedores dentro de un bloque (coordenada BAROTI) con el fin de minimizar los reacomodamientos. Utilizando algoritmos genéticos para el primer objetivo y un algoritmo tipo *greedy* para la segunda.

Este proyecto no tomará en cuenta otros actores que intervienen el patio de contenedores, por lo que se debe suponer que los recursos siempre son suficientes. Por ejemplo, se asume que existen suficientes camiones para el transporte de contenedores que permitan el tráfico inmediato de un contenedor, ya sea de importación o exportación. La formulación de este proyecto no se aplicará sobre un terminal específico, pero se tomarán en cuenta varios aspectos operativos de algunos terminales estudiados, en general del STI, Chile. Para las pruebas se definirá un terminal teórico con ciertas características, similares al Puerto de Arica, ya que los datos de prueba serán extraídos de ese terminal.

Una de las medidas de calidad del servicio de un terminal portuario es el tiempo de permanencia (*dwell time*) de una nave en el puerto. Mientras menos permanezcan las naves en los muelles, más naves se pueden atender aumentando los ingresos. Esto también es positivo para la atención a las empresas navieras, que desean permanecer el menor tiempo posible, mejorando así el servicio. En *A genetic algorithm to solve the storage space allocation problem in a container terminal* [Bsj08] se ve que el balance de las cargas de trabajo de los bloques funciona muy bien para minimizar el tiempo de permanencia de una nave. En el trabajo mencionado anteriormente solo se considera el flujo de importación ya que este flujo representa el 85% en el terminal considerado (*Shahid Rajaei Terminal*, Irán). En este caso, se tomarán en cuenta solo los flujos de exportación, ya que para la importación, se van despachando de forma relativamente rápida después de la descarga, asumiendo un sistema de citas, similar al que se usa en el terminal STI. En este caso, a medida que los contenedores se descargan se van despachando ya que se cita al cliente para retirar el contenedor.

Un problema específico de patio es la presencia de movimientos innecesarios de contenedores (Aquellos movimientos que no tienen como fin la remoción del mismo contenedor del patio, sino que despejar la vía para la remoción de otro contenedor). Esto sucede cuando en un momento dado se requiere de la remoción de un contenedor que está bajo otros. Si hubiese una buena planificación se podría evitar tener que cambiar de lugar un contenedor, lo que tiene un costo altísimo debido a la gran potencia requerida para levantar un el peso de un contenedor. Se debe considerar que un contenedor estándar de 20 pies podría llegar a las ~26,8 toneladas. Para resolver esto, se tiende a segregar los contenedores según la

nave en que se embarcarán y el puerto de destino (además de otras segregaciones para el tipo de contenedor, peso, etc.), los contenedores de cada segregación deben estar juntos dentro de los espacios, así se evita la posibilidad de que ocurra este problema, aunque se pierda un poco de espacio. Sin embargo, este proyecto considerará contenedores individuales, de este modo se aprovecha mejor el espacio, pero aparece el problema de la presencia de los movimientos innecesarios. Por esta razón, se considera dentro de los objetivos el minimizar estos movimientos.

Otro problema específico de patio tiene que ver con su recurso principal, el espacio. Generalmente los puertos tienen suficiente espacio para almacenar contenedores. En el caso contrario, cuando la disponibilidad de espacio peligra, una opción, es ampliar el lugar dedicado para este propósito, aunque no siempre es posible. Cuando la demanda se acerca a la capacidad del puerto y la ampliación de espacios no es posible, se debe hacer una planificación considerando el espacio como recurso fundamental. Este problema no se da con tanta frecuencia por lo que existe una pequeña cantidad de publicaciones que considera el espacio como recurso a optimizar. Al tomar contenedores individuales se aprovecha mejor el espacio.

El patio estará dividido en bloques. En este caso, cada bloque estará dividido según el sistema BAROTI. La cantidad de bloques en este puerto teórico y la cantidad de *bays*, *row* y *tiers* por bloque, es relevante solo para las pruebas. Los movimientos de contenedores serán efectuados por grúas tipo gantry (RMG o RGT), por lo que cada bloque estará equipado con una o más de estas grúas. Para el muelle se considerarán distintos sitios de atraque donde son asignadas las embarcaciones.

Como ya se ha visto en *The Yard Allocation Problem* [Cfl02] se toman en cuenta solicitudes de unidades abstractas de espacio, las cuales pueden aumentar mientras pasa el tiempo hasta que se cumple el tiempo final de la solicitud. Pero no considera segregación alguna. En este proyecto no se incluirán segregaciones para tipos de contenedores. La razón para no considerar contenedores *reefer* es que no es un problema decidir en qué bloque deben ir ubicados, ya que es natural asignarlos a aquel bloque destinado para tales efectos. La restricción de almacenamiento de contenedores vacíos tiene que ver con el apilamiento de contenedores. Esto no se considerará ya que los contenedores serán mezclados en los *stack*. Además debe considerarse el peso del contenedor en la estiba del barco, por lo que en el patio deberán estar apilados en el orden inverso a cómo deberían ir en el barco. En este caso la clasificación por peso será por diez categorías, donde deberán ir ordenados por niveles en la estiba. La razón para no considerar cargas peligrosas, es que según el estándar IMO para el manejo de cargas peligrosas [Cam04], existen 17 segregaciones y 289 restricciones recomendando la distancia entre contenedores de estos tipos. Estas recomendaciones son excesivas.

En *The Yard Allocation Problem*, se asume el conocimiento de la demanda presente y futura con toda la información, la que se usa para planificar los espacios. Esto en la vida real es imposible. En general se conoce la información relativamente próxima, y con eso se debe planificar. Por esta razón se ha optado por un enfoque de horizonte rodante, tal como en diversos trabajos, donde se conoce la información en un horizonte de cierta cantidad de períodos de tiempo, y se toman las decisiones para el período presente. Esto se hace período a período. A modo de ejemplo, se planifica día a día, teniendo en cuenta un futuro próximo. En este caso el horizonte tomará 10 períodos los que representan 5 días, donde cada período representa 12 horas. Esto considerando que en el terminal STI los contenedores de exportación llegan al puerto hasta 3 días antes de la llegada del barco.

Un cuello de botella ocasionado por la mala planificación de espacios, puede llegar a retrasar el servicio de carga/descarga de un barco, lo que podría afectar en gran medida las planificaciones de sitios de atraque del puerto, además de las planificaciones que tengan las navieras para sus barcos. La planificación del muelle está fuera del alcance del proyecto, por lo que se optará por no medir los tiempos de permanencia, ya que al aumentar más de lo planificado puede que un barco deba ser atendido mientras se atiende a otro. En este caso, se tendrá como objetivo el balance de las cargas de trabajo de los sectores o bloques del patio de contenedores.

## 4.1 Definición formal de la solución

En base a lo anterior, a continuación se define formalmente el problema en dos etapas, cada una con su objetivo, la solución final integra estos dos niveles. Para cada período a planificar se ejecutarán los algoritmos correspondientes los siguientes niveles.

**Nivel 1:**

Se usará un algoritmo genético que determinará en qué bloque debe ubicarse cada uno de los contenedores de tal manera que se logre minimizar la diferencia de las cargas de trabajo de cada sector, minimizar la distancia de los traslados de los contenedores desde el bloque hacia el muelle correspondiente. Indirectamente el algoritmo afecta la cantidad de reacomodos, por lo que también se añade a la función objetivo.

**Nivel 2:**

Ya establecida la asignación de contenedores a bloques, se debe determinar en qué espacio *baroti* debe ir ubicado cada contenedor, de tal forma de minimizar los reacomodos.

Estos reacomodos tienen que ver con la estiba del barco. Para este caso, se utilizará un algoritmo del tipo *greedy*.

### 4.1.1 Supuestos y definiciones

Existen suficientes recursos para el tránsito entre *quay cranes* y el patio de contenedores (camiones), el tiempo para estos efectos es despreciable.

Las grúas de patio son del tipo pórtico, ya sean *rail mounted gantry cranes* o *rubber tired gantry cranes*, pueden mover contenedores a una tasa fija constante ingresada como parámetro.

La tasa de las grúas RMG incluye movimientos tanto de recepción como para embarques. Esto se considera para calcular las cargas de trabajo.

Las unidades de tiempo son atómicas, nada pasa entre ellas. Los barcos llegan en el tiempo estimado entregado en la planificación naviera.

Una vez ubicado un contenedor, este no puede cambiar de posición hasta que deba ser embarcado (Excepto en casos de reacomodo y solo dentro de un bloque).

Solo existen contenedores de 20 pies (1 TEU), del tipo normal. Los contenedores tienen diez categorías de peso.

La estiba del barco se hará ordenadamente según los criterios de puerto de destino y peso, teniendo este último mayor relevancia para mantener el balance del barco.

El patio del puerto posee una cantidad de bloques (yardas), cada una con recursos fijos (TEUs por unidad de tiempo) para mover contenedores.

Para el muelle se considera una cantidad parametrizada de sitios de atraque.

La diferencia entre las distancias de los distintos bloques con el sitio de embarque se considera para tomar decisiones, para esto se extraen datos de la distribución del terminal.

Cada vez que se deba remover un contenedor de un bloque para ser embarcado, se deben contar las interferencias, se cuenta uno por cada contenedor que esté encima del contenedor a remover. No se considerará el total de movimientos innecesarios.

Se considerarán viajes, los cuales relacionan un barco, su sitio asignado y el tiempo de llegada. Cada contenedor tiene pre-asignado un viaje en el cual se debe embarcar.

### 4.1.2 Parámetros de entrada

Para este modelo se deben tener en cuenta los siguientes parámetros correspondientes a la información ingresada previamente a una base de datos:

- Planificación naviera:
    - Nave.
    - Viaje.
    - ETA.
    - Fecha.
    - Sitio del muelle.
- Distribución del puerto:
    - Cantidad de bloques.
    - Tamaño de cada bloque (Cantidad de *bays*, *row* y *tiers*).
    - Distancia entre bloques y sitios.
- Container Assigment List (CAL):
    - Contenedor.
    - Peso.
    - Puerto destino.
    - Viaje.

### 4.1.3 Objetivo

Para el nivel 1 se debe determinar la ubicación dentro del patio de cada contenedor que llega en el período de planificación

Para el nivel 2 se debe determinar la coordenada BAROTI en la que cada contenedor debe ser ubicado dentro del bloque al cual fue asignado en el nivel 1.

El objetivo es minimizar tres aspectos importantes:

- La diferencia de cargas de trabajo entre bloques. Peso determinado por $\alpha$ . Este aspecto se ve directamente afectado por el nivel 1.

- Las distancias entre cada contenedor almacenado en el patio y el respectivo sitio donde será embarcado. Peso determinado por $\beta$ . Este aspecto se ve directamente afectado por el nivel 1.

- La cantidad de interferencias encontradas al extraer el contenedor de su respectivo bloque para transportarse al respectivo sitio de atraque. Peso determinado por $\gamma$ . Este aspecto se ve directamente afectado por el nivel 2, sin embargo se ve indirectamente afectado por el nivel 1.

Sea:

$C_{it}$: Número de contenedores que debe procesar el bloque i en el período t.

$d_i$: Distancia entre el bloque asignado al contenedor i y el sitio del muelle correspondiente.

$I_i$: Número de interferencias detectadas para sacar el contenedor i del patio de contenedores.

$$\min Z = \alpha \sum_{t=1}^{T} \left( \max_{i \in B} C_{it} - \min_{i \in B} C_{it} \right) + \beta \sum_{i=1}^{C} d_i + \gamma \sum_{i=1}^{C} I_i$$

**Procedimiento de evaluación**

Considerando que la variable de decisión es el bloque en que se almacenará cada contenedor, calcular las cargas de trabajo consiste en sumar los flujos de entrada y salida del bloque en un período de tiempo. La distancia bloque-muelle correspondiente a un contenedor está dada en la distribución del terminal, entregado como parámetro. La cantidad de interferencias al retirar un contenedor de un bloque, será calculada por el algoritmo de la etapa dos.

### 4.1.4 Algoritmo Genético nivel 1

Como se dijo anteriormente se usará un algoritmo genético como motor de generación y selección de soluciones. A continuación se describe el algoritmo genético propuesto, el cual será sometido a prueba con distintas variaciones para probar la utilidad de los algoritmos genéticos para resolver el problema formulado en este capítulo.

## Genotipo

En esta etapa, el algoritmo genético debe determinar en qué bloque debe ir ubicado cada uno de los contenedores que lleguen en el período de planificación, por lo que cada posible solución es representada por un individuo a través de un genotipo compuesto por números enteros.

$$G = C_1\ C_2\ C_3\ C_4\ C_5\ C_6 \ldots\ \ldots C_n$$

La posición de cada gen representa un contenedor. El valor de cada gen representa el bloque en que será almacenado el contenedor, por lo que su valor debe fluctuar entre los identificadores de bloques, según sea el caso.

## Generación de población inicial

El objetivo en este caso es crear una población lo más dispersa posible para abarcar uniformemente el espacio de búsqueda. Por ello, se crean cromosomas cuyos genes serán generados aleatoriamente dentro del rango de posibilidades, sin importar si los valores de cada gen se repiten. El tamaño de la población dependerá de la cantidad de contenedores correspondientes al período a evaluar.

Durante el procedimiento de generación de la población inicial podría ocurrir la violación de la restricción de capacidad de bloque. Al momento de la generación del genotipo no se puede determinar este hecho ya que el costo de evaluar esto es altísimo, por lo que se optará por reparar el genotipo.

## Selección

Para la selección se usará el mecanismo de selección por torneo, explicado en detalle en el capítulo 3. Se formaran parejas de a dos sucesivamente hasta completar una cantidad de torneos determinada, donde en cada torneo se seleccionará el individuo más apto (menor dwell time). La cantidad de torneos se ira ajustando en las pruebas observando su comportamiento, comenzando con un valor base de 1. La aptitud del individuo o genotipo se evaluará bajo los términos detallados en la sección 4.1.4.

## Cruzamiento

En este caso se utilizará cruzamiento de un punto (ver capítulo 3), donde se selecciona un punto al azar, y se intercambian las porciones del cromosoma después del punto de corte. Las parejas sujetas a cruzamiento también serán seleccionadas aleatoriamente. La probabilidad de cruzamiento se irá determinando en las pruebas según su comportamiento, con un valor base de 90%.

La aplicación de este operador podría comprometer la factibilidad de la solución, incluso cuando los padres sean factibles, por lo que se requiere de un mecanismo de reparación.

**Mutación**

Para la mutación, cada gen de cada individuo tiene una probabilidad de ser mutado. Si un gen es seleccionado para la mutación, se toma un par de genes aleatorios y se intercambian, esto no genera conflicto de factibilidad.

**Criterio de término**

Dado que el valor óptimo se desconoce, el algoritmo genético se detendrá al cabo de cierto número de generaciones. Este número se determinará durante las pruebas al estudiar la convergencia del algoritmo genético. Otra alternativa es alcanzar cierta homogeneidad en la población.

### 4.1.5 Algoritmo Nivel 2

Ya establecidas las asignaciones en el nivel 1, donde se asignó cada contenedor a un bloque, se debe determinar la posición dentro del bloque de forma de minimizar los movimientos innecesarios.

Para el caso de la nivel 2, no se define formalmente una función objetivo, sin embargo se calculará el número de interferencias. Se describe un algoritmo tipo *greedy* para determinar la posición dentro del bloque, este tiene como objetivo minimizar el número de interferencias. El procedimiento es el siguiente, para cada bloque en cada período de planificación.

Ya determinado el conjunto de contenedores asignados a cada bloque, se toma un contenedor y se toma como primer candidato el primer *row* del primer *bay* (el cual está más cercano a la posición del camión). Se avanza por *row* hasta llegar al final del *bay*, luego se va al siguiente *bay*. Si no hay contenedor alguno en esa posición, se coloca allí. De estar lleno, la posición se descarta (*stack* completo). Si hay espacio se evalúa el embarque, si fuese distinto, se coloca en esa posición solo si el contenedor(a asignar) debe ser retirado antes, sino se descarta la posición. Si el contenedor corresponde al mismo embarque, se evalúa su categoría de peso. Si tiene un peso menor, se coloca en la posición, de lo contrario se descarta la posición. Si al pasar por todas las posiciones no se ha colocado el contenedor, entonces se hace una nueva pasada, y se colocará el contenedor en el primer espacio que encuentre disponible.

De este sencillo modo se intenta disminuir los movimientos innecesarios con respecto a una asignación aleatoria.

# 5 Sistema de pruebas

Dada la naturaleza única de la formulación, es imposible la comparación de resultados con otros trabajos, sin embargo, sí se utilizaron datos reales como entrada. Estos datos fueron ingresados a una base de datos, para su posterior procesamiento. Los datos provienen de un terminal portuario real en Chile, el Terminal Puerto Arica. El set de datos describe las distintas actividades del puerto relacionadas con la entrada y salida de carga, que ocurrieron entre el mes de enero y julio del año 2012. De esto se extrajo datos relevantes para el proyecto. Para el proyecto se requerían datos de los contenedores, específicamente: un identificador del contenedor, peso, tiempo de llegada, y el embarque correspondiente. Para eso se procesaron los datos brutos para extraer los datos deseados y eliminar datos que no cumplían con las exigencias nombradas anteriormente, por ejemplo contenedores que no tenían embarques asignados.

Para evaluar el funcionamiento del algoritmo genético, se ejecutaron alrededor de 1000 instancias, estudiando cómo se comporta cada uno de los parámetros en el resultado final de la función objetivo.

El algoritmo fue desarrollado con el entorno de desarrollo Visual Studio 2012 en el lenguaje C#. Las pruebas se efectuaron en una máquina con procesador Intel Core i7 2670qm con 8gb de memoria principal, sobre Windows 8 Professional.

## 5.1 Pruebas de sensibilidad de parámetros genéticos

A continuación se presentan una serie de pruebas previas evaluando valores de distintos parámetros. En todas las pruebas se consideran los siguientes pesos: $\alpha = 1$ , $\beta = 100$ , $\gamma = 100$ .

### 5.1.1 Prueba 1, Evaluación del tamaño de la población

En esta prueba se evaluaron 300 instancias con distintos tamaños de población, considerando los siguientes parámetros fijos:

- Probabilidad de cruza: 90%
- Probabilidad de mutación: 10%

- Número de generaciones: 100

En esta prueba de consideraron tamaños de poblaciones desde 10 individuos hasta 200 individuos, de diez en diez. Se evaluaron 15 instancias por cada uno.

#### 5.1.1.1Resultado

Como se ve en el siguiente gráfico de dispersión (figura 5.1), donde cada punto representa el resultado de una instancia, el algoritmo muestra una a mejorar los resultados a medida que se aumenta la población. El tiempo de ejecución es directamente proporcional al tiempo de ejecución del algoritmo, por lo que se estimó un máximo de 200 para no aumentar en demasía el tiempo de pruebas. Por la misma razón se mantendrá un tamaño de población de 100 para las siguientes pruebas previas, reservándose el tamaño de población 200 para las pruebas finales.

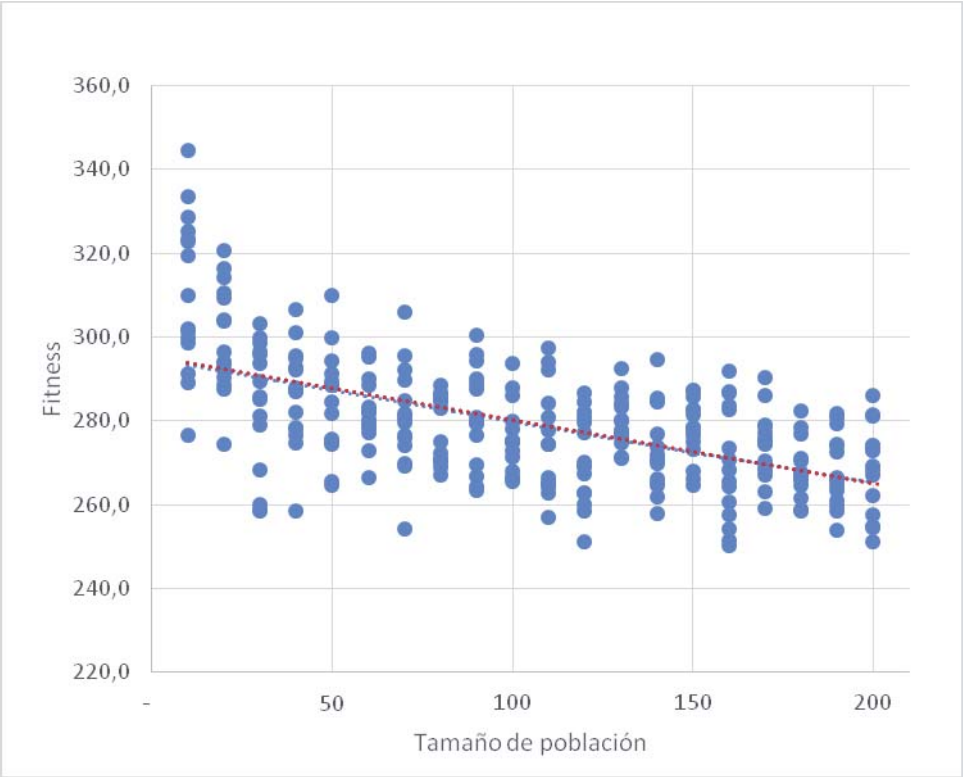


**Figura 5.1 - Resultado prueba 1**

### 5.1.2 Prueba 2, Evaluación de la probabilidad de mutación y cruza

En esta prueba se evaluará el comportamiento de la probabilidad de mutación en un rango de 0,05 hasta un 0,3 (en saltos de). Se evaluó en salto de a 0,05. Además para cada uno de estos valores se evaluó en conjunto de un valor de probabilidad de cruza en un rango de 0,5

hasta un 0,9. Lo anterior en saltos de 0,1. Más adelante se efectuarán pruebas más finas para la probabilidad de cruza. En total se evaluaron alrededor de 180 instancias, unas 6 veces cada conjunto de parámetros. Los siguientes parámetros permanecen fijos:

- Tamaño de población: 100
- Número de generaciones: 100

#### 5.1.2.1Resultado

Como se ve en el siguiente gráfico (figura 5.2), una alta probabilidad de mutación afecta de manera negativa el rendimiento del algoritmo. Se puede apreciar que mientras más bajo, mejores resultados se obtienen. No obstante se debe mantener un valor mayor a cero, ya que es necesario mantener una pequeña cantidad de capacidad exploratoria aleatoria para evitar la convergencia en un óptimo local. Se mantendrá el valor 0,05 para la probabilidad de mutación para el resto de las pruebas.

Además se observa que los mejores resultados consideran probabilidades de cruza entre 0,8 y 0,9. En la siguiente prueba se considera este rango, pero con saltos menores.

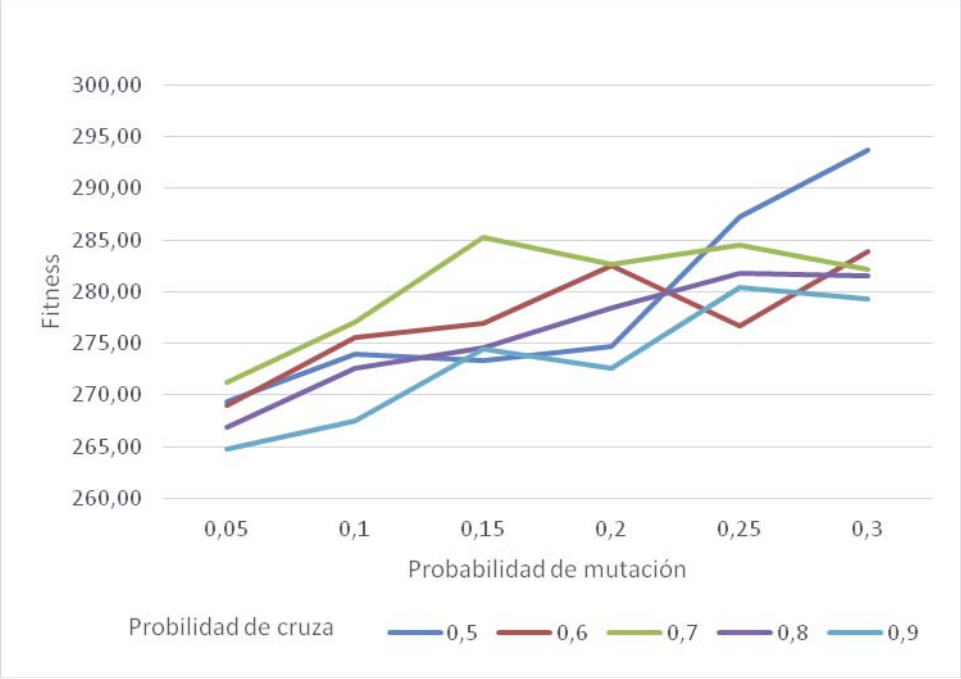


**Figura 5.2 - Resultado prueba 2**

### 5.1.3 Prueba 3, Evaluación de la probabilidad de cruza

En esta prueba se evalúa el comportamiento de la probabilidad de cruza de modo más fino, en un rango de 0,8 – 0,9 con saltos de 0,01. Se ejecutaron unas aproximadamente unas 100 instancias, 10 por cada valor de probabilidad de cruza. Se dejaron fijos los siguientes parámetros:

- Tamaño de población: 100
- Número de generaciones: 100
- Probabilidad de mutación: 5%

#### 5.1.3.1Resultado

Como se puede observar en el siguiente gráfico de dispersión (figura 5.3), no existe una tendencia tan clara. Sin embargo para las pruebas finales se establece el valor 0,88 como probabilidad de cruza.

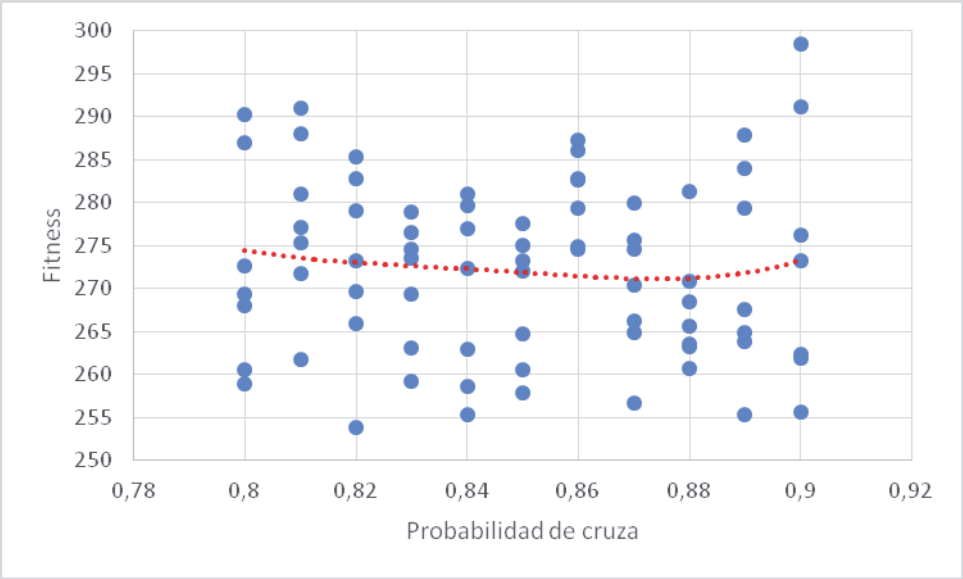


**Figura 5.3 - Resultado prueba 3**

## 5.2 Pruebas Finales

En el análisis de sensibilidad de los parámetros se desprende información sobre el comportamiento del algoritmo genético a medida que los parámetros cambian. Con esta

información se crearon tres escenarios de pruebas. Cada escenario consiste en una determinada ocupación máxima. Se determinaron los escenarios con cargas de un 10%, 60% y 100%. Los escenarios de fueron puestos a prueba con los siguientes parámetros:

- Población: 200 individuos.
- Generaciones: 500 - 1000.
- Probabilidad de cruza: 80% - 99%.
- Probabilidad de mutación: 0,5%.
- Mecanismo de Selección: Ruleta.
- Mecanismo de cruzamiento: Single point.
- Mecanismo de mutación: Intercambio.

Las pruebas que se presentan a continuación corresponden a las pruebas finales. Primero se presentan pruebas individuales para cada factor a optimizar (balance de cargas, distancia bloque-muelle y movimientos extras) con ocupación de un 60%. Con esto se ajustaron los pesos $\alpha$, $\beta$, $\gamma$ para que tuvieran una importancia equivalente en el resto de las pruebas. Esto se hace por la falta de ponderadores de importancia reales, por ejemplo, el costo.

Luego con los ponderadores ajustados se ejecutaron tres pruebas adicionales, las que difieren en su respectivo nivel de ocupación.

### 5.2.1 Prueba 4: Solo balance de carga con 60% de ocupación ($\alpha = 1$)

En este caso, la prueba consiste en probar solo el balance de cargas. Se utilizó el escenario de 60% de ocupación en su momento peak. Como se puede apreciar en la figura 5.4, después de la generación N°100 no hay una mejora mantenida del fitness (menos es mejor). Lo que indica que fue relativamente sencillo encontrar una solución aceptable. Se llegó a un buen balance de cargas. Este factor depende directamente del algoritmo genético.

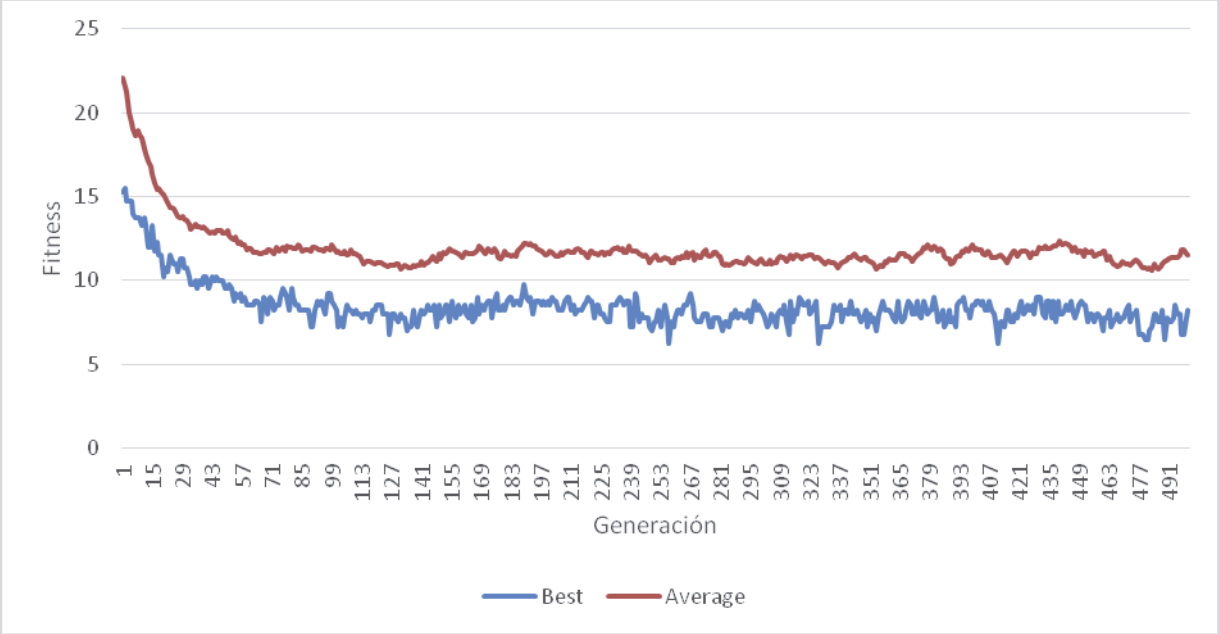


**Figura 5.4 - Resultado prueba 4**

### 5.2.2 Prueba 5 Solo distancia contenedor-muelle con 60% de ocupación (β = 1)

En este caso, la prueba consiste en probar solo la distancia contenedor-muelle. Se utilizó el escenario de 60% de ocupación en su momento peak. Como se puede apreciar en la figura 5.5, hay una disminución sostenida del fitness (menos es mejor) hasta la generación N°500. Se aprecia una convergencia en el final de la prueba. Este factor depende directamente del algoritmo genético.

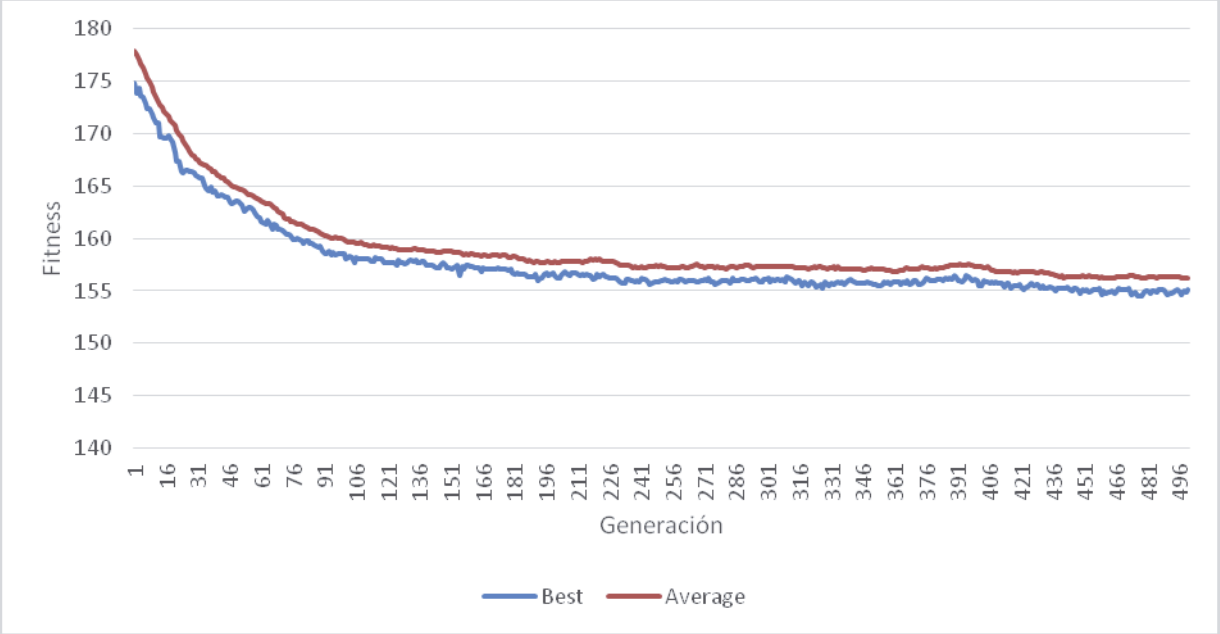


**Figura 5.5 - Resultado prueba 5**

### 5.2.3 Prueba 6: Solo movimientos extras con 60% de ocupación (γ = 1)

En este caso, la prueba consiste en probar solo los movimientos extras. Se utilizó el escenario de 60% de ocupación en su momento peak. Como se puede apreciar en la figura 5.6, hay una disminución sostenida del fitness (menos es mejor) hasta la generación N°500. Se aprecia una convergencia en el final de la prueba. Intuitivamente este factor no depende del algoritmo genético ya que la planificación dentro del bloque la realiza el algoritmo de la sección 4.1.5, pero aun así la prueba indica que la planificación macro si influye indirectamente en la disminución de los movimientos extras.

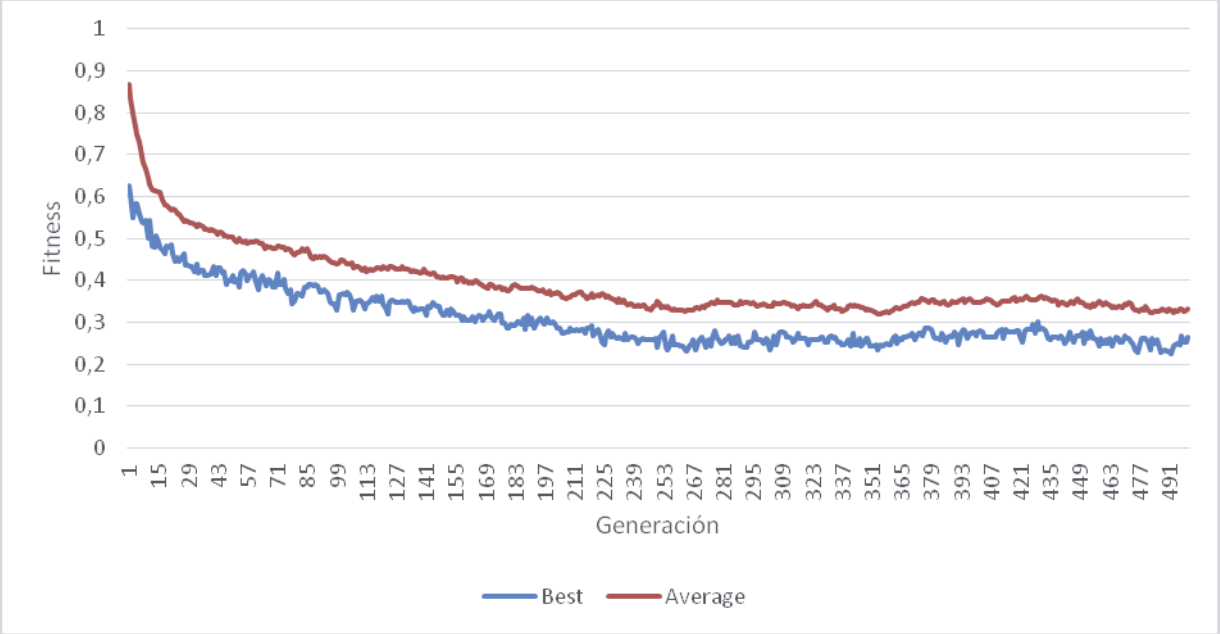


**Figura 5.6 - Resultado prueba 6**

### 5.2.4 Prueba 7: Pesos ajustados con 60% de ocupación (α = 1, β = 23, γ = 750)

En este caso, la prueba consiste en probar todos los factores a optimizar, con la información del a prueba anterior se establecieron los ponderadores como aparecen en el título. Se utilizó el escenario de 60% de ocupación en su momento peak. Como se puede apreciar en la figura 5.7, hay una disminución sostenida del fitness (menos es mejor) hasta la generación N°500. Se aprecia una convergencia en el final de la prueba.

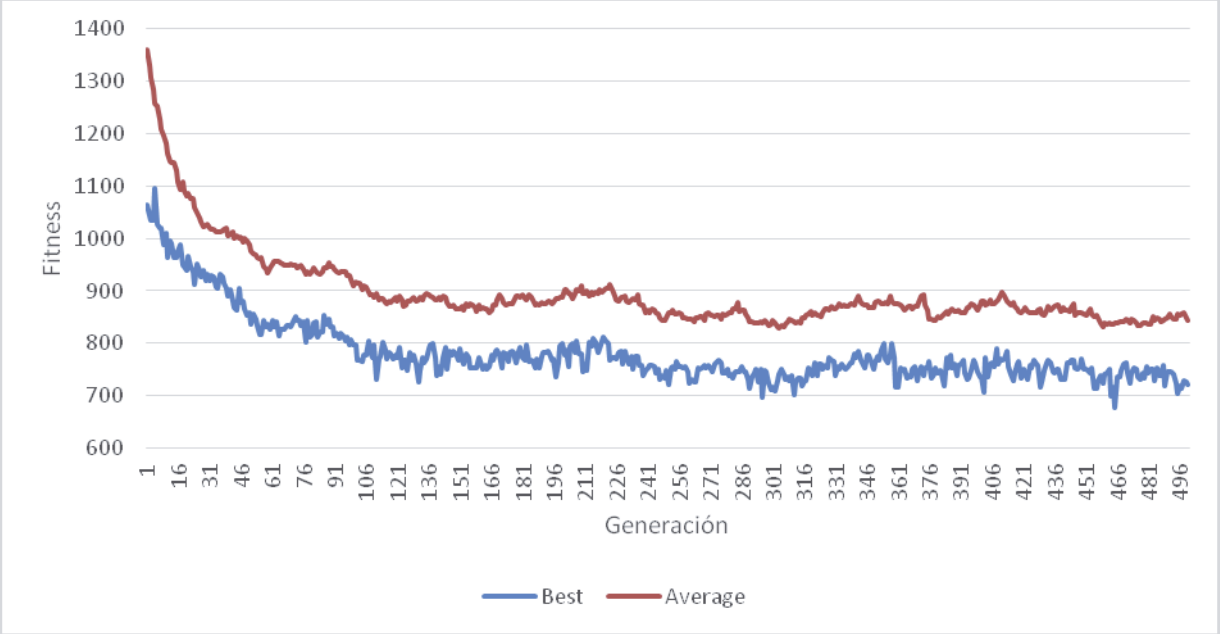


**Figura 5.7 - Resultado prueba 7**

### 5.2.5 Prueba 8: Pesos ajustados con 10% de ocupación (α = 1, β = 23, γ = 750)

En este caso, la prueba consiste en probar todos los factores a optimizar, con la información del a prueba anterior se establecieron los ponderadores como aparecen en el título. Se utilizó el escenario de 60% de ocupación en su momento peak. Como se puede apreciar en la figura 5.8, hay una disminución sostenida del fitness (menos es mejor) hasta la generación N°500. Se aprecia una convergencia en el final de la prueba. Se aprecia solo una pequeña diferencia en los valores de fitness con respecto al escenario anterior de 60% de ocupación.

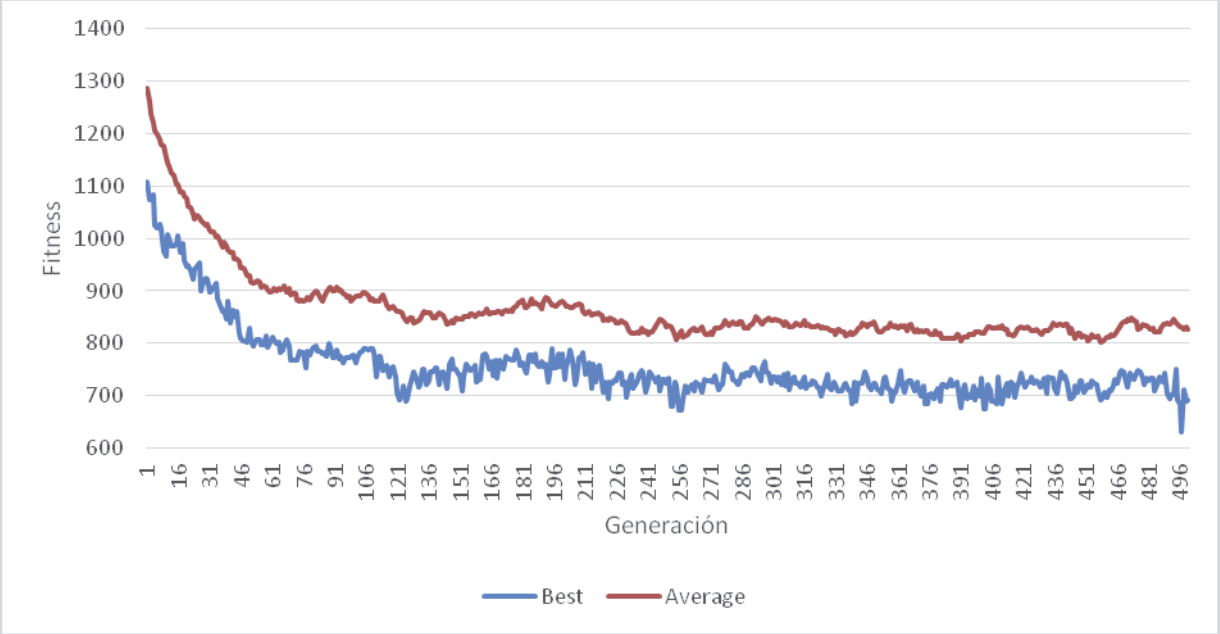


**Figura 5.8 - Resultado prueba 8**

### 5.2.6 Prueba 9: Pesos ajustados con 100% de ocupación ($\alpha = 1$, $\beta = 23$, $\gamma = 750$)

En este caso, la prueba consiste en probar todos los factores a optimizar, con la información del a prueba anterior se establecieron los ponderadores como aparecen en el título. Se utilizó el escenario de 100% de ocupación en su momento peak. Como se puede apreciar en la figura 5.7, hay una disminución sostenida del fitness (menos es mejor) hasta la generación N°500. Se aprecia una convergencia en el final de la prueba. A diferencia de los escenarios más holgados (10% y 60% de ocupación), la curva de fitness es más alta, lo cual es normal para un escenario tan difícil como este, donde existe una ocupación completa del patio de contenedores.

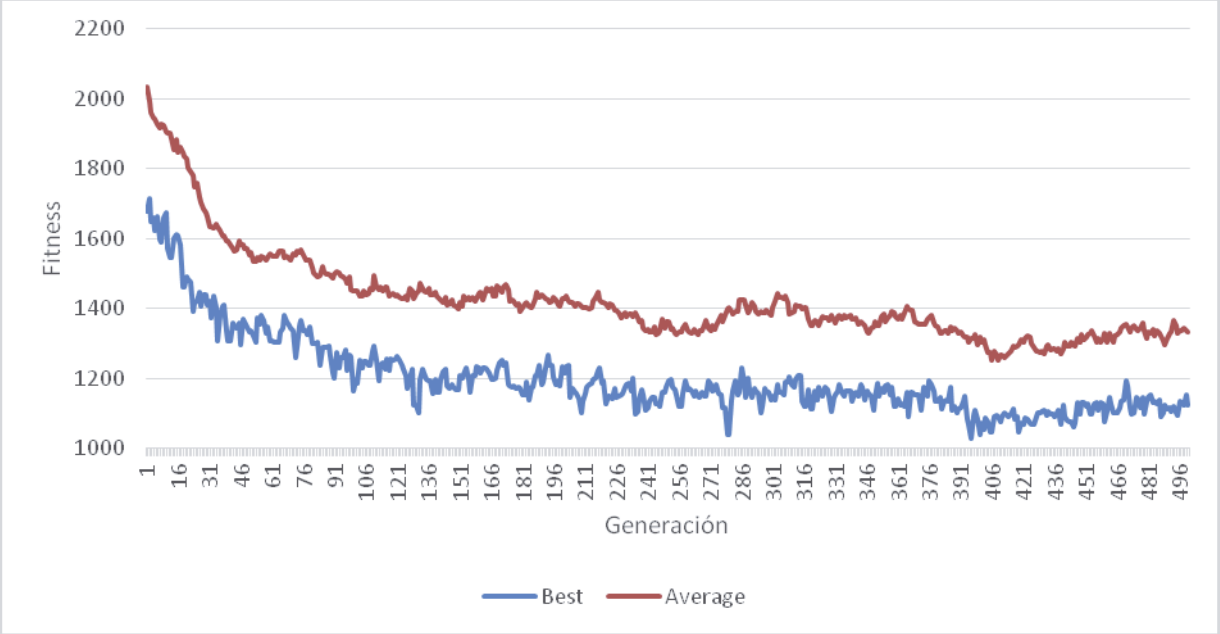


**Figura 5.9 - Resultado pruebas 9**

# 6 Conclusiones

En este informe ha presentado lo que ha trabajado para el proyecto de título del autor. En un principio se determinaron los objetivos generales y específicos del presente proyecto. Se estudió el problema a resolver y la técnica: los algoritmos genéticos, para elaborar un marco teórico que permitió tomar ciertas decisiones con respecto al proyecto. Para ello se revisaron trabajos relacionados con el problema, para poder entender cuanto se ha trabajado en ello, junto con sus ventajas y desventajas. Para los algoritmos genéticos, en un principio se estudió el algoritmo genético original, propuesto por Holland. Luego se estudiaron trabajos sobre algoritmos genéticos aplicados a distintos problemas, para comprender como afectaban las distintas técnicas de algoritmos genéticos en ciertos tipos de problema.

El problema a resolver tiene una formulación matemática sencilla, pero su resolución es compleja, debido a que primero se debe plantear una solución para la evaluación de la función objetivo además de usar algoritmos genéticos que generen y evalúen las soluciones al problema mismo. Algunos operadores pueden causar problemas. Por ejemplo es muy probable que un cruzamiento clásico genere soluciones inválidas. Lo mismo para con la mutación. Por esa razón se diseñaron mecanismos que reparen la solución para evitar el incumplimiento de alguna de las restricciones.

Se propuso una solución de dos niveles, en la primera se realiza la asignación a nivel de bloque, la segunda realiza las asignaciones a nivel atómico, entregando finalmente la planificación completa del patio de contenedores.

Se realizaron una gran cantidad de instancia de pruebas para encontrar el mejor conjunto de parámetros para dar con la mejor solución posible. En las pruebas se demostró la efectividad del algoritmo genético y su potencial uso en el área.

A pesar de que el algoritmo genético no planifica a nivel interno del bloque, si influye indirectamente, y si es capaz de optimizar la cantidad de movimientos extras que se dan al remover un contenedor cubierto por otro u otros contenedores.

Aplicar una solución de este tipo en el mundo real es complicado ya que se requiere de información completa con suficiente tiempo de anticipación como para que la planificación sea de alguna utilidad. Estas condiciones no se cumplen en la mayoría de los casos, ya que las operaciones portuarias dependen de muchos actores. Es por eso que este tipo de soluciones deben complementarse con otros sistemas que soporten cambios en los eventos relacionados con la actividad portuaria.

# Referencias


[Hol75] John Henry Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Harbor, Michigan, 1975.

[Gol89] David Edward Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Massachusetts, 1989.

[Bsj08] Mohammad Bazzazi, Nima Safaei, Nikbakhsh Javadian, *A genetic algorithm to solve the storage space allocation problem*, Computers & Industrial Engineering, Volume 56, Issue 1, 2009.

[Cfl02] Ping Chen, Zhaohui Fu ,Andrew Lim, *The Yard Allocation Problem*, National University of Singapore, AAAI-02 , Edmonton, Alberta, Canada, 2002.

[Cfl02-1] Ping Chen, Zhaohui Fu, Andrew Lim, *Using Genetic Algorithms To Solve The Yard Allocation Problem*, AAAI-02, Edmonton, Alberta, Canada, 2002.

[Cflr03] Ping Chen, Zhaohui Fu, Andrew Lim, Brian Rodrigues, *The General Yard Allocation Problem*, GECCO 2003, Chicago.

[www1] URL: *http://www.singaporepsa.com*.

[Can98] Erick Cantú-Paz, *A Survey of Parallel Genetic Algorithms*, CALCULATEURS PARALLELES, Volume 10, 1998.

[Sri94] M. Srinivas, Lalit M, Patnaik, *Genetic Algorithms: A Survey*, IEEE Computer Society, Computer, Volume 26, Issue 6, 1994.

[Schb92] Nicol N. Schraudolph, Richard K. Belew, *Dynamic Parameter Encoding for Genetic Algorithms*, Machine Learning, Volume 9, Issue 1, 1992.

[Jam87] James Baker, *Reducing Bias and Ineffieciency in the Selection Algorithm*, Second International Conference on Genetic Algorithms and their application, Hillsdale, New Jersey, 1987.

[Gol95] David Edward Goldberg, Brad Miller, *Genetic Algorithms, Tournament Seleccion, and Effect of Noise*, Complex Systems, Volume 9, 1995.

[Elo11] Julián Alberto Elorrieta Decombe, *Planificación y gestión de operaciones portuarias en patio de contenedores*, Escuela de Ingeniería Industrial, Pontificia Universidad Católica de Valparaíso, Chile, 2011.

[Lim98] Andrew Lim. *The Berth Planning Problem*. Operations Research Letters, Volume 22, Issues 2-3, 1998.

[Cam04] João Gilberto Campos. *Curso de Capacitação para Interpretação de Plano de Carga para Navios – Full Container*, INCATEP, Brasil, 2004.

[Cota12] Rodrigo Covarrubias, Francisco Tapia, *Desarrollo, implementación y aplicación de un enfoque multi-objetivo para la planificación de operaciones de un patio de contenedores*, Escuela de Ingeniería Industrial, Pontificia Universidad Católica de Valparaíso, Chile, 2012.

[web2] URL: *http://www.subtrans.gob.cl/subtrans/organismos/portuarias.html*

[web3] URL: *http://www.containerhandbuch.de/chb_e/stra/index.html?/chb_e/stra/stra_01_03_03.html*