Industrial Data
ISSN: 1560-9146
iifi@unmsm.edu.pe
Universidad Nacional Mayor de San Marcos
Perú

Ruiz Lizama, Edgar
Optimización multi-objetivo al problema de distribución de planta usando algoritmos genéticos: cuestiones previas para una propuesta de solución
Industrial Data, vol. 17, núm. 2, julio-diciembre, 2014, pp. 120-137
Universidad Nacional Mayor de San Marcos
Lima, Perú

Revista de la Facultad de Ingeniería Industrial
17(2): 120-137 (2014) UNMSM
ISSN: 1560-9146 (Impreso) / ISSN: 1810-9993 (Electrónico)

Optimización multi-objetivo al problema de distribución de planta usando algoritmos genéticos: cuestiones previas para una propuesta de solución

RECIBIDO: 15/07/14 ACEPTADO: 12/10/14

EDGAR RUIZ LIZAMA*

RESUMEN

El artículo es una revisión del estado del arte y de los conceptos teóricos necesarios para elaborar una propuesta de solución al Facility Layout Problem FLP. En el artículo se examinan conceptos tales como meta heurísticas, computación evolutiva, programación evolutiva, algoritmos genéticos, tendientes a plantear una alternativa de solución al difícil problema de la distribución en planta.

Palabras clave: Problema de Distribución de planta, meta heurísticas, optimización, algoritmos genéticos

MULTI-OBJECTIVE OPTIMIZATION TO THE PROBLEM OF DISTRIBUTING PLANT USING GENETIC ALGORITHMS: PRELIMINARY ISSUES FOR A PROPOSED SOLUTION

ABSTRACT

The article reviews the state of the art and theoretical concepts needed to develop a proposed solution to the Facility Layout Problem FLP. Article concepts such as meta heuristics, evolutionary computation, evolutionary programming, genetic algorithms, aimed to propose an alternative solution to the difficult problem of plant distribution are examined.

Keywords: Facility Layout Problem, meta heuristics, optimization, genetic algorithms

INTRODUCCIÓN

En los momentos actuales, las organizaciones, independientemente de su tamaño y del sector de actividad, han de hacer frente a un mundo globalizado en los que han de conciliar la satisfacción de sus clientes con la eficiencia económica de sus actividades. Los problemas de Distribución de Planta son un grupo de problemas de optimización consistentes en la participación de una región plana de dimensiones conocidas (generalmente rectangular) en departamentos de área conocida, de tal manera que se minimice el costo asociado con las instalaciones previstas entre dichos departamentos, y es por ello que hace uso de técnicas de la Investigación de Operaciones o IO. El propósito de la presente investigación, es estudiar el problema de la Distribución de Planta para proponer una solución tal que aplique la optimización Multi-Objetivo de una organización, teniendo como herramienta de solución los algoritmos genéticos.

1. SITUACIÓN PROBLEMÁTICA

Un elemento clave en muchos sistemas de producción y servicios, es el diseño de las instalaciones. El Problema de la Distribución de Planta (PLD, de Plan Layout Design Problem o FLP de Facility Layout Problem), es concerniente con encontrar el arreglo espacial **óptimo** de una colección de lugares o instalaciones en un espacio determinado.

Este problema, es el que se enfrentan las Gerencias de Gestión de Operaciones y las de Distribución. La distribución de planta determina la ordenación de los medios productivos. Muther [Muther et al., 1979] plantea con el sistema SPIF (Systematic Planning of Industrial Facilities) la planta industrial como un sistema compuesto por cinco subsistemas físicos interrelacionados entre ellos y con el entorno: Sistema Distribución de Planta, Sistema de Manutención-Almacenaje, Sistema Edificio, Sistema Instalaciones, Sistema Comunicaciones. Debido al gran número de factores a considerar, lograr la distribución de planta de manera eficiente no es una tarea sencilla si no, todo lo contrario, pues una planta industrial es un subsistema complejo en el que interactúan máquinas, materiales y hombres que se sirven de un conjunto de instalaciones.

* Magister en Informática PUCP, Ingeniero Industrial UNMSM. Docente Principal del DAISI, Facultad de Ingeniería Industrial UNMSM. E-mail: eruizl@unmsm.edu.pe

"Los problemas de distribución de planta son un grupo de problemas de optimización consistentes en la participación de una región plana de dimensiones conocidas (generalmente rectangular) en departamentos de área conocida, de tal manera que se minimice el costo asociado con las instalaciones previstas entre dichos departamentos. Estos costos pueden deberse al transporte (incluyendo costos asociados con la construcción de los sistemas de manutención) o preferencias entre departamentos" [Tate *et al.,* 1995].

"El problema de la distribución de planta consiste en determinar la distribución más eficiente de un número de departamentos indivisibles con requerimientos de área desigual en el interior de una instalación. El objetivo es minimizar los costos del transporte de materiales dentro de la planta considerando dos grupos de restricciones: los requerimientos de área y las restricciones de localización de los departamentos (no pueden solaparse, deben ser colocados en el interior de la planta, y algunas pueden necesitar una localización fija o no pueden ser colocados en regiones específicas" [Meller *et al.,* 1996].

"El problema de la distribución de planta consiste en la disposición física de un número dado de departamentos o máquinas con una configuración determinada. En el contexto de las máquinas manufactureras, el objetivo es minimizar el costo del transporte de materiales requeridos entre los diferentes departamentos" [Mavridou *et al.,* 1997]

"El problema de la distribución de planta consiste en localizar la disposición óptima de un grupo de instalaciones sujetas a restricciones cualitativas o cuantitativas" [Shayan *et al.,* 2004]

Desafortunadamente los tomadores de decisiones que han planteado soluciones óptimas a muchos problemas en gestión de operaciones, han encontrado que hay una clase de problemas que son extremadamente difíciles de resolver. Estos problemas en la literatura se denominan *NP-hard*, por sus siglas en inglés (Nom Polinomial hard) es decir "Problemas No Polinomiales Difíciles". Entre ellos se encuentran: la secuenciación de proyectos (project scheduling), localización y distribución de instalaciones o facilidades o FLP (facilities location and layout problem).

2. MARCO TEÓRICO

La distribución o disposición del equipo (instalaciones, máquinas, etc.) y áreas de trabajo, es un problema ineludible para todas las plantas industriales y comerciales; no es posible evitarlo. El objetivo en distribución de plantas (o en plantas) es hallar una ordenación de las áreas de trabajo y del equipo, que sea la más económica para el trabajo, al mismo tiempo que la más segura y satisfactoria para los empleados (Muther, 1997).

2.1. INVESTIGACIONES RELACIONADAS

Koopmans y Beckmann (1957) fueron los pioneros al modelar los Problemas de Distribución de Plantas FLP´s (Facility Layout Problems) de área igual definiéndolo como un problema de asignamiento cuadrático QAP (Quadratic Assignment Problem) en el cual se minimizan los costos del manejo de materiales, que resultan de ubicar cada máquina, en cada una de las localizaciones posibles. El trabajo significó el punto de partida para los investigadores, muchos de los cuales destacan la importancia de los QAP en los problemas de distribución de plantas de áreas iguales. Según Garey y Johnson (1979), y Sanhi y Gonzalez (1976), este es un problema del tipo NP-Hard.

La experiencia real muestra que en los sistemas de manufactura, los departamentos tienen áreas desiguales. Armour y Buffa (1963) fueron los primeros en proponer un método de intercambio de parejas para los FLP de área desigual. Asumiendo que los departamentos tienen forma rectangular Tong (1991) plantea la ubicación de los departamentos en bahías, aplicando la estructura de bahía flexible definida como una representación continua la cual permite que los departamentos se localicen en bahías paralelas con anchos diferentes. Hernández Gress E.S. et. al. (2009) proponen un algoritmo genético para el diseño optimo de aéreas desiguales con al menos nueve departamentos. Wong y komarudin (2010) le dan especial importancia a esta representación. García Hernández L. *et. al.* (2013) proponen un algoritmo genético interactivo para el FLP de aéreas desiguales.

Tam (1992 a,b) propuso utilizar la estructura de árbol de corte STS (Slicing Tree Structure) para modelar aquellas instalaciones que tienen forma rectangular y geometría flexible, esto quiere decir que el ratio entre el largo y el ancho del área asignada al departamento puede variar dentro de ciertos márgenes. Mas, et. al.(2006) proponen un indicador que mide la capacidad de un árbol de corte para generar soluciones geométricamente aceptables para los FLP's basados en la estructura de árbol de corte.

Drira, Pierreval y Hajri-Gabouj (2007) realizan un interesante estudio y clasificación de los FLP's,

incluyendo la formulación y las técnicas de solución a este tipo de problemas.

Ardestani *et al.* (2009) aplican para el FLP programación entera mixta y se consideran dos objetivos comúnmente en conflicto: minimizar el costo de manipulación de materiales entre departamentos y maximizar el ratio de la cercanía entre ellos. Aiello, La Scalia, y Enea (2012) proponen un Algoritmo Genético Multi-Objetivo MOGA(Multi Objetive Genetic Algorithm) para el FLP en aéreas desiguales basado en una estructura de árbol de corte, donde las ubicaciones relativas de las facilidades en el terreno se representan en una matriz de ubicación codificada en dos cromosomas. Aiello, La Scalia y Enea (2013) proponen un ranking no dominado como un MOGA y el método electre para los FLP´s de área desigual.

Nawaz *et al.* (2013) proponen un enfoque evolutivo para resolver el FLP multi-objetivo de áreas desiguales usando la variable de búsqueda del vecino más cercano VNS (Variable Neighborhood Search) con un esquema adaptativo que presenta los layouts finales como un conjunto de soluciones Pareto-optimas. El VNS es un método exploratorio de búsqueda local cuya idea básica es el cambio de la vecindad dentro de una búsqueda local.

2.2. BASES TEÓRICO-CIENTÍFICAS

2.2.1. Distribución de plantas

La distribución en plantas implica la ordenación física de los elementos industriales. Esta ordenación, incluye, tanto los espacios necesarios para el movimiento del material, almacenamiento, trabajadores indirectos y todas las otras actividades o servicios, como el equipo de trabajo y el personal de taller (Muther, 1997).

La distribución de planta consiste en seleccionar el arreglo más eficiente de las instalaciones físicas, con el fin de lograr eficiencia al combinar los recursos para producir un artículo o servicio (James Tompkins, 2007).

Existen diferentes tipos de distribución de plantas, una clasificación clásica (Muther, 1997) se basa en el desplazamiento del material.

El tipo de distribución en puesto fijo, es aquel donde el material no se desplaza, son los operarios los que van hacia el producto con las máquinas portátiles necesarias para hacer las distintas operaciones e incorporar componentes al producto. Fabricación de barcos, aviones, entre otros.

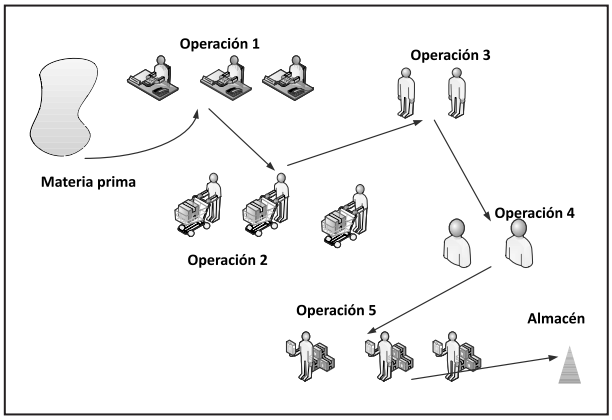
El tipo de distribución por proceso o denominada funcional, donde la planta se organiza en secciones especializadas, por tipos de máquinas. Todas las máquinas que realizan el mismo tipo de proceso o función se agrupan formando una sección: tornos, fresadoras, taladros, pintura, entre otros. Ver figura 2.1.

Una vez acabadas las operaciones en una sección, el material es trasladado al centro de trabajo, donde se tiene que realizar la siguiente operación, quedando en espera junto a otros tipos de piezas para entrar en las máquinas correspondientes, formando así una cola.

La distribución en línea o por producto, también denominada por producción en cadena, un producto o un tipo de producto se realiza en un área, contrario a la distribución fija, el material está en movimiento. Esta disposición dispone cada operación inmediatamente al lado de la siguiente. Es decir que el proceso que se lleve a cabo, está ordenado de acuerdo a una secuencia de las operaciones.

La ventaja de la distribución por función es la mejor ocupación de las máquinas, puesto que estas trabajan independientemente, al máximo de sus posibilidades de producción, y una vez acabada la orden de fabricación de una pieza, se preparan para la siguiente.

Figura 2.1. Distribución por proceso.



Fuente: Elaboración propia.

La distribución de planta, se aplica a la selección de la disposición de las instalaciones físicas no sólo de las fábricas, sino también de las oficinas, hospitales, aeropuertos, centros comerciales y todo tipo de instalación. Una denominación más precisa es la “distribución de las instalaciones” (J. Tompkins, 2007). El objetivo general de la distribución de

plantas es maximizar la eficiencia de los recursos humanos y materiales.

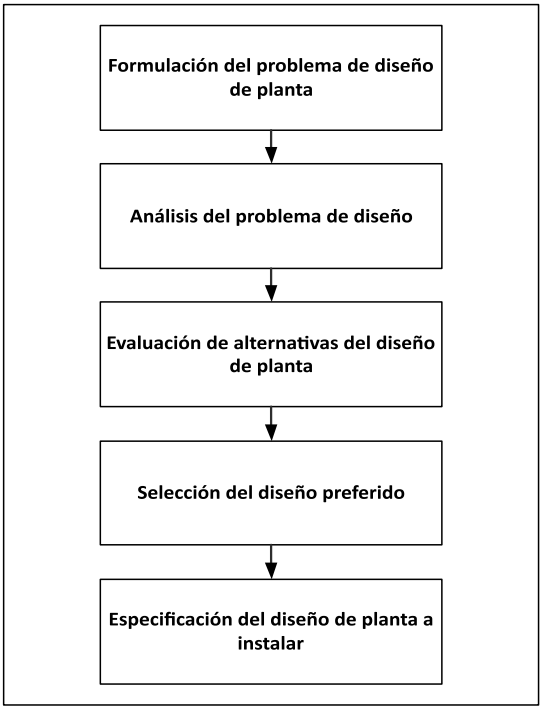
Para conseguir este objetivo, J. Velasco (2010) plantea que es necesario:

- Optimizar la producción en el mismo espacio para reducir así los costos de desplazamiento de materiales, alquiler, mantenimiento y limpieza.
- Reducir transportes, con el consiguiente ahorro de mano de obra.
- Reducir las esperas en el curso del proceso productivo para disminuir el volumen de inmovilizado en curso y el plazo de fabricación.
- Buenas condiciones de trabajo tanto desde el punto de vista tecnológico (la ergonomía) como psicológico (el ambiente).

El proceso del desarrollo de la distribución de plantas (en inglés *plant layout*) contiene ambos elementos: de arte y ciencia (Francis, White, 1992). Lo de arte es dependiente de la creatividad, síntesis y un estilo en el diseño de plantas; lo de ciencia por utilizar el análisis, reducción y la deducción.

Como todo proceso de diseño de ingeniería; la distribución de plantas consiste de una serie de pasos, entre los que destacan, la formulación, análisis y evaluación, entre otros. Ver la figura 2.2.

Figura 2.2. Procedimiento en la distribución de planta

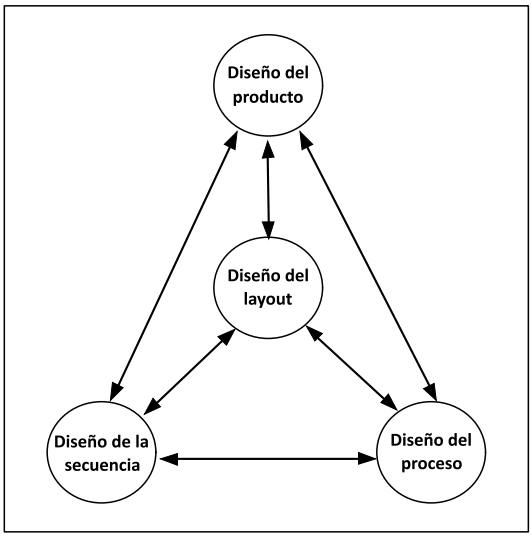


Fuente: Elaboración propia.

El problema del layout de plantas, puede ser un problema muy complejo de diseño de sistemas; y el objetivo es probar a encontrar una solución satisfactoria que cumpla con los propósitos de las componentes del sistema (Francis, White, 1992).

Las componentes del sistema layout, se encuentran en el diseño del producto, diseño de la secuenciación y el diseño del proceso. Ver la figura 2.3.

Figura 2.3. Comunicación de las componentes en layout



Fuente: Elaboración propia (de acuerdo a Francis, White, 1992).

Según Francis y White (1992), uno de los criterios comúnmente usados para evaluar alternativas de layout es el costo del manejo de materiales. Existiendo una serie de metas importantes.

Uno de los objetivos según Francis y White (1992) puede ser:

- Minimizar la inversión en equipos.
- Minimizar el tiempo de producción.
- Utilizar el espacio existente en forma efectiva.
- Minimizar el costo del manejo de materiales.
- Facilitar el proceso de manufactura, entre otros.

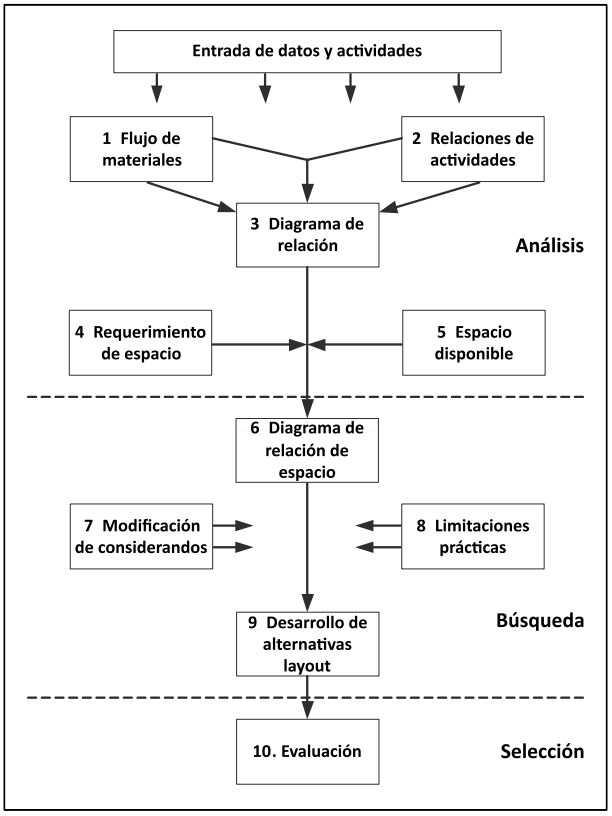
En adición a estos objetivos, existen una serie de restricciones a la solución, como regulaciones locales, parámetros ambientales y prevención de riesgos laborales.

Un número de procedimientos han sido desarrollados para facilitar el diseño layout. Nadler (1961) presenta un sistema ideal con las recomendaciones de Immer, Apple y Reed.

Muther (1961) desarrolla el Planeamiento Sistemático Layout o SLP (de *Systematic Layout Planning*), procedimiento que ha sido aplicado a producción, transporte, almacenamiento, servicios, actividades de oficina, entre otros. EL SLP es simplemente un camino ordenado para ejecutar el proceso layout.

El procedimiento SLP se inicia con el acopio de datos y actividades y se plantea el diagrama de relación. Teniendo el espacio requerido y el disponible se construye el diagrama de relación de espacios. El desarrollo de las alternativas de distribuciones se consigue con las consideraciones y limitaciones prácticas. El SLP considera tres áreas: Análisis, Búsqueda y Selección. Ver la figura 2.4.

Figura 2.4. Procedimiento SLP



Fuente: Elaboración de acuerdo a Francis y White, 1992.

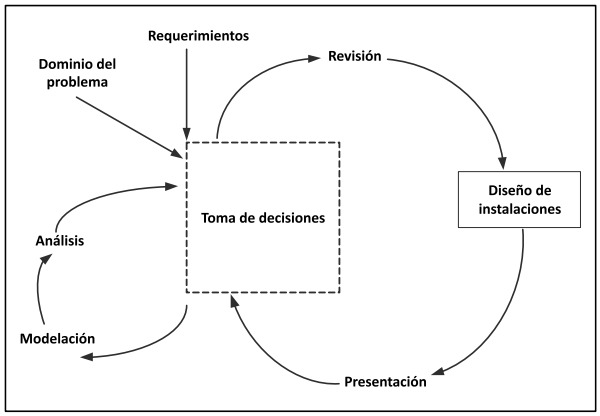
2.2.2. Planeamiento de la distribución de plantas

El planeamiento de la distribución de plantas, envuelve decisiones concernientes con el diseño de la manufactura, donde converge un equipo multidisciplinario de diseñadores y planificadores, siendo la comunicación y la documentación dos áreas críticas (Francis y White, 1992).

El diseño layout es dependiente del planeamiento y control de la producción, producto, requerimiento de los procesos, requerimiento de máquinas, diseño de almacenamientos, agrupamiento de máquinas, entre otros.

El proceso de la distribución de plantas es un proceso de información basado en el dominio del problema, y donde la toma de decisiones se interacciona con la modelación y el análisis, a partir de los requerimientos de la organización. Ver la figura 2.5.

Figura 2.5. Proceso del layout de instalaciones



Fuente: Elaboración de acuerdo a Francis y White, 1992.

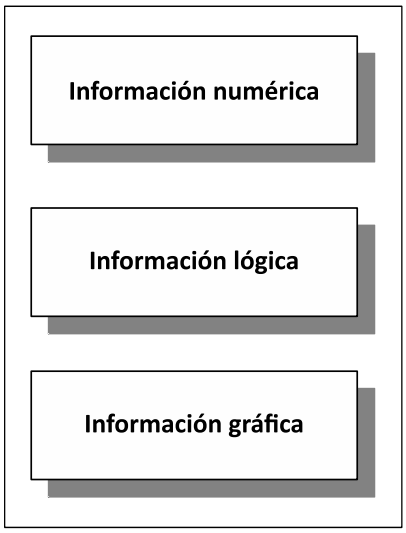
A menudo los planificadores necesitan emplear modelos matemáticos layout en la búsqueda de buenas decisiones o evaluar layout tentativos.

Según Francis y White (1992), existen tres tipos de información en el planeamiento layout: la numérica, que está relacionada con el espacio requerido por cada actividad o el flujo total entre dos actividades; la lógica, que describe las preferencias en las instalaciones; y finalmente la gráfica que está referida al dibujo o esqueleto de la planta. Ver la figura 2.6.

Según la información gráfica, los bloques de una planta representan áreas para cada una de las actividades. Esta información es importante para modificar el layout existente. En la figura 2.7, se observa el área de cada actividad y las relaciones existentes entre cada una de ellas en una instalación. La modificación se logra reordenando, intercambiando la posición y cambiando la forma de cada actividad.

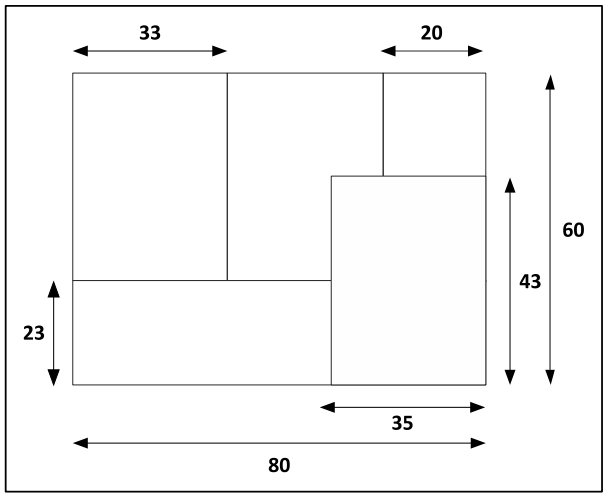
El proceso mecánico de la modificación del layout, se logra mediante el uso de algoritmos adecuados, que mediante procesos computacionales automatizan las operaciones del planeamiento layout.

Figura 2.6. Información en el planeamiento layout



Fuente: Elaboración propia.

Figura 2.7. Bloque layout

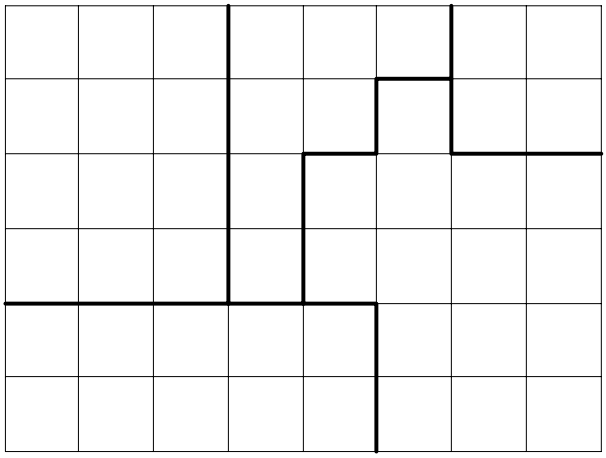


Fuente: Elaboración de acuerdo a Francis y White, 1992.

En la figura 2.8, se presenta la aproximación de los bloques dentro del área de las instalaciones. La información numérica vista como una representación matricial del bloque layout se presenta en la figura 2.9.

La representación matricial, un arreglo ordenado de filas y columnas, contiene celdas que en su intersección corresponden a la unidad de medida de un área unitaria. Esta representación permite el intercambio de áreas unitarias correspondientes de una actividad con otra actividad.

Figura 2.8. Aproximación de los bloque layout



Fuente: Elaboración de acuerdo a Francis y White, 1992.

Figura 2.9. Representación matricial

1	1	1	2	2	2	3	3
1	1	1	2	2	4	3	3
1	1	1	2	4	4	4	4
1	1	1	2	4	4	4	4
5	5	5	5	5	4	4	4
5	5	5	5	5	4	4	4

Fuente: Elaboración de acuerdo a Francis y White, 1992.

Los criterios usados por los planeadores en layout, están basados en la teoría de la utilidad. Tomando la teoría de la utilidad se asume el vector de atributos X_i para un layout L_i con la utilidad $u = f(X_i)$. En la comparación L_i y L_j, si $u_i > u_j$, entonces el layout L_i es preferido a L_j.

La función utilidad ha sido aproximada a un modelo de puntaje o score, siendo la forma del modelo score layout la siguiente:

$$s = g(X)$$

En la literatura se reportan modelos score layout del tipo:

- Basados en la adyacencia.
- Basados en la distancia.

- Basados en la ponderación distancia y adyacencia.

El modelo score basado en adyacencia, está basado en el ranking de las adyacencias, haciendo uso de seis clases de adyacencia y X_i el número de adyacencias en la clase respectiva. El software ALDEP fue implementado usando este modelo de score.

Sean las seis clases de adyacencia (*A, E, I, O, U y X*) con pesos de 64, 16, 4, 1, 0 y -1024 respectivamente, se utilizan para calificar las adyacencias entre cada par de actividades. La función score es:

$$s = \sum_{i=1}^{6} w_i X_i$$

Un largo score indica mejor layout.

El modelo score basado en la distancia, aproxima el flujo entre para de actividades desde el costo C_{ij} . La distancia D_{ij} entre cada par de actividades (comúnmente con el supuesto rectilíneo), determina el modelo score basado en la distancia:

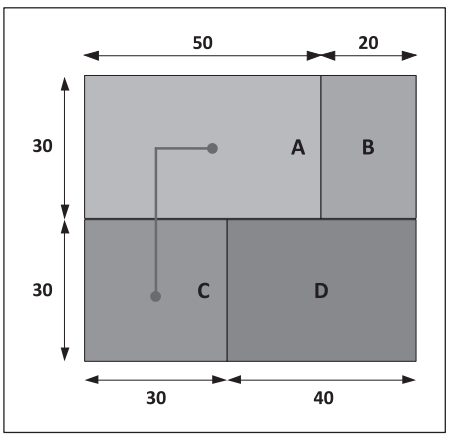
$$s = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} c_{ij} D_{ij}$$

El cálculo de la distancia rectilínea o de Manhattan, es la medida de las distancias de los segmentos entre los centroides. En la figura 2.10 los centroides de las actividades *A* y *B* son y respectivamente. La distancia rectilínea entre *A* y *B* es la suma de los segmentos entre los centroides, es decir .

La fórmula de la distancia entre los puntos *A* y *B* con coordenadas y , viene dada por:

$$D_{AB} = d(P_A, P_B) = |x_A - x_B| + |y_A - y_B|$$

Figura 2.10. Distancia rectilínea o de Manhattan



Fuente: Elaboración propia.

El modelo score basado en la ponderación distancia y adyacencia, combina la distancia rectilínea con las clases de adyacencia, en la función score:

$$s = \sum_{i=1}^{6} \sum_{j=1}^{6} w_{ij} X_{ij}$$

El software CORELAP fue implementado usando este último modelo score.

Los modelos score descritos, son la visualización del SLP, desde la óptica de los bloques layout, y hacen uso de simples operaciones aritméticas (Francis y White, 1992).

2.2.3. Metaheurística

Una jerarquía de problemas ha aparecido, asociado con un conjunto de técnicas para su solución. Estos problemas, son denominados como de programación matemática; donde se busca el valor de , tal que cumple las siguientes condiciones:

$$\min f(\boldsymbol{x})$$
Sujeto a:
$$g_j(\boldsymbol{x}) \geq b_j, j = 1, \ldots, m$$
$$h_k(\boldsymbol{x}) = \mathbf{0}, k = 1, \ldots, p$$

donde *f, g* y *h,* son funciones del parámetro X ϵ R^n. Las técnicas para resolver tales problemas son siempre iterativas, por naturaleza, y su convergencia es estudiada usando matemáticas del mundo real.

Para problemas no restringidos o con restricción de igualdad, el cálculo diferencial, encuentra el óptimo, ya que el problema es buscar los valores extremos de la función.

La complejidad computacional estudia el esfuerzo o costo de la resolución de un problema. El esfuerzo necesario para resolver un problema de forma eficiente puede variar grandemente.

En la resolución de un problema, se busca la mejor solución entre un conjunto de posibles soluciones. Al conjunto de todas las posibles soluciones a un problema concreto se le denomina espacio de búsqueda.

La búsqueda de una solución, se reduce a buscar el valor extremo (mínimo o máximo) en el espacio de búsqueda. Este espacio de búsqueda a veces puede ser bien definido, pero en la mayoría de las ocasiones sólo se tiene el conocimiento de algunos puntos en el espacio de búsqueda.

Al planteamiento de un problema concreto, se encuentra una serie de algoritmos que se pueden

aplicar. Se suele decir que el orden de complejidad de un problema es la del mejor algoritmo que se conozca para resolverlo. Es así como se clasifican los problemas y los estudios sobre algoritmos que se aplican a la realidad.

En el tiempo han aparecido diversos estudios, que han llevado a la constatación de que existen problemas muy difíciles, problemas que desafían incluso la utilización de las computadoras para resolverlos.

La cuestión es que existen, por una parte, problemas resolubles de manera determinista mediante algoritmos polinomiales y en un tiempo polinomial, como puede ser, por ejemplo la resolución de ecuaciones, la realización de sumas, productos, etc., pudiendo acotar el tiempo de resolución, más o menos largo, de una manera aceptable. Estos son los denominados problemas P (de Polynomial).

Los algoritmos de complejidad polinomial, se dice que son tratables, en el sentido de que suelen ser abordables en la práctica. Los problemas para los que se conocen algoritmos con esta complejidad, se dice que forman la clase P. Aquellos problemas para los que la mejor solución que se conoce es de complejidad superior a la polinomial, se dice que son problemas intratables. Sería interesante encontrar alguna solución polinomial (o mejor) que permitiera abordarlos.

Sin embargo, también existen problemas NP que pueden resolverse de forma no determinista, probando una solución conjeturada. Esta comprobación es de una gran rapidez en comparación con el tiempo polinomial necesario en general para la resolución determinista de los problemas P.

Algunos de estos problemas intratables, pueden caracterizarse por el curioso hecho de que puede aplicarse un algoritmo polinomial, para comprobar si una posible solución es válida o no. Esta característica lleva a un método de resolución no determinista consistente en aplicar heurísticos para obtener soluciones hipotéticas que se van desestimando (o aceptando) a ritmo polinomial. Los problemas de esta clase se denominan NP (la N de no-deterministas y la P de polinomial).

Se conoce una amplia variedad de problemas de tipo NP, de los cuales destacan algunos de ellos de extrema complejidad. Gráficamente se puede decir que algunos problemas se hayan en la “frontera externa” de la clase NP. Siendo estos los peores problemas posibles de la clase NP. Estos problemas se caracterizan por ser todos “iguales” en el sentido de que si se descubriera una solución P para alguno de ellos, esta solución sería fácilmente aplicable a todos ellos.

Se puede afirmar que los problemas fáciles están en P (y en NP), pero los difíciles de verdad, *sólo* están en NP y se llaman NP-completos. Es más, si se descubriera una solución para los problemas NP-completos, esta sería aplicable a todos los problemas NP y, por tanto, la clase NP desaparecería del mundo científico al carecerse de problemas de ese tipo.

Una alternativa para resolver los problemas NP-completos son las denominadas metaheurísticas (como son los algoritmos genéticos). Ejemplos de problemas NP-completos son el problema del agente viajero Traveller Salesman Problem(TSP), el problema del coloreamiento de un grafo, etc. Estos problemas de optimización combinatoria, están caracterizados por contar con un espacio de solución finito, siendo muy frecuentes en la vida diaria y en el diseño de ingeniería.

En la actualidad, todos los algoritmos conocidos para problemas NP-completos utilizan tiempo exponencial, con respecto al tamaño de la entrada. Se desconoce si hay algoritmos más rápidos, por lo cual, para resolver un problema NP-completo de tamaño arbitrario, se utiliza uno de los siguientes enfoques: aproximación, probabilísticos, heurísticos, entre otros.

Las aproximaciones Metaheurísticas suelen ser utilizadas en problemas de este tipo. Un ejemplo de algoritmo heurístico de complejidad O(n log n) es el algoritmo voraz, utilizado para la coloración de vértices, en el diseño y construcción de algunos compiladores.

Desde su origen, la programación matemática, se encuentra abocada a problemas para los que no existe método analítico alguno que permita obtener, con seguridad y en un tiempo conveniente, el óptimo teórico. Éste es, por ejemplo, el caso de los problemas combinatorios en que el sentido común da por imposible la enumeración. Es más que normal que el tamaño y la naturaleza de ciertos problemas combinatorios nos prohibían abordarlos por la vía del sentido común. Nuestro buen sentido y razón, educados por la ciencia, saben distinguir particularmente los problemas NP completos, para los cuales no existe un algoritmo que en tiempo polinomial sea capaz de encontrar la solución.

La investigación de operaciones ha establecido, por las razones expuestas, métodos denominados heurísticos, que no proporcionan el óptimo formal, pero susceptibles de llegar a soluciones buenas, tanto más fiables en cuanto permiten determinar

al mismo tiempo una cota (superior o inferior) del óptimo teórico con el que se comparan. Con el auge de las PC, hacia principios de la década de 1980, estos métodos han ido ganando terreno, puesto que se iba haciendo, cada vez más, factible y fácil intentar diferentes heurísticas y juzgar su eficacia relativa.

Durante los últimos años han aparecido una serie de técnicas modernas o no tradicionales. Estas son conocidas como metaheurìsticas, cuya finalidad es la de encontrar buenas soluciones a problemas de optimización. Entre ellas se pueden enumerar los algoritmos genéticos, el recocido simulado, la búsqueda tabú, redes neuronales, etc. Su aplicación a los problemas de secuenciación de todo tipo es una finalidad típica y clásica. Es más, prácticamente todas ellas están basadas en intentar resolver, de la mejor forma posible, problemas típicos de Organización de la Producción. Así, los problemas típicos de secuenciación de trabajos en máquinas, de asignación de rutas, planificación de la producción, etc. han sido, son y, casi con toda seguridad, serán el banco de pruebas de las más modernas técnicas de búsqueda de soluciones a problemas en los que, de entrada, se sacrifica la posibilidad de encontrar la solución óptima.

El término metaheurística o meta heurística, fue acuñado por F. Glover en 1986. Con ello, pretendía "definir un procedimiento maestro de alto nivel, que guía y modifica otras heurísticas, para explorar soluciones más allá de la simple optimalidad local".

A partir de la definición de F. Glover, se encuentran en las literaturas, otras definiciones alternativas de metaheurística o heurística moderna.

Una metaheurística, es un método de solución general, que proporciona tanto una estructura general, como criterios estratégicos para desarrollar un método específico que se ajuste a un tipo particular del problema.

Una clasificación jerárquica para intentar una taxonomía a las metaheurìsticas, consiste en armar un árbol desde el punto de vista conceptual. Existen heurísticas trayectoriales y heurísticas poblacionales.

Entre las metaheurìsticas trayectoriales destacan, las basadas en búsqueda local (como Búsqueda Tabú o TS (Tabu Search) y Recocido Simulado o SA), búsqueda iterativa (como Búsqueda Local Iterativa o ILS (Iterative Local Search) y Búsqueda por entorno adaptativo borroso o FANS) y búsqueda multi-arranque (como Procedimientos de búsqueda miope, aleatorizada y adaptativa o GRASP). Por otro lado, entre las metaheurìsticas poblacionales, destacan, las basadas en combinación de soluciones como Algoritmos Genéticos o GA y Búsqueda Dispersa o SS (Sparse Search) y las basadas en movimientos (como ACO, de Optimización de Colonias de Hormigas y SI, de Inteligencia de Enjambres).

Los Algoritmos Genéticos o GA (del inglés *Genetic Algorithms*) fueron introducidos por Holland para imitar algunos de los mecanismos que se observan en la evolución de las especies. Los mecanismos de la genética no son conocidos en profundidad pero sí algunas de sus características: la evolución ocurre en los cromosomas; un ser vivo da vida a otro, mediante la decodificación de los cromosomas de sus progenitores, el cruce de los mismos, y la codificación de los nuevos cromosomas formando los descendientes; las mejores características de los progenitores se trasladan a los descendientes, mejorando progresivamente las generaciones.

Los algoritmos de recocido simulado o SA (de *Simulated Annealing*) fueron introducidos por Kirkpatrick en 1983, para la optimización de problemas combinatorios con mínimos locales. Utilizan técnicas de optimización no determinista: no buscan la mejor solución en el entorno de la solución actual sino que generan aleatoriamente una solución cercana y la aceptan como la mejor si tiene menor costo, o en caso contrario con una cierta probabilidad *p*; esta probabilidad de aceptación irá disminuyendo con el número de iteraciones y está relacionada con el empeoramiento del costo.

Estos algoritmos derivan de la analogía termodinámica con el proceso metalúrgico del recocido: cuando se enfría un metal fundido suficientemente despacio, tiende a solidificar en una estructura de mínima energía (equilibrio térmico); a medida que disminuye la temperatura, las moléculas tienen menos probabilidad de moverse de su nivel energético; la probabilidad de movimiento se ajusta a la función de Boltzmann.

Entre los distintos métodos y técnicas heurísticas de resolución de problemas combinatorios surge, en un intento de dotar de "inteligencia" a los algoritmos de búsqueda local, tal como el algoritmo de búsqueda tabú, de Glover y Laguna en 1997.

La búsqueda tabú o TS de *Tabù Search*, a diferencia de otros algoritmos basados en técnicas aleatorias de búsqueda de soluciones cercanas, se caracteriza porque utiliza una estrategia basada en el uso de estructuras de memoria para escapar de los óptimos locales, en los que se puede caer al "moverse" de una solución a otra por el espacio de soluciones. Al igual que en la búsqueda local, la búsqueda tabú selecciona de modo agresivo el

mejor de los movimientos posibles en cada paso. Al contrario que sucede en la búsqueda local, se permiten movimientos a soluciones del entorno aunque se produzca un empeoramiento de la función objetivo, de manera que sea posible escapar de los óptimos locales y continuar estratégicamente la búsqueda de mejores soluciones.

Numerosos algoritmos basados en el principio de Darwin, han empezado a desarrollarse sobre las últimas tres décadas. Ellos están usando el término de "métodos de computación evolucionaria".

El término de algoritmos evolucionarios o evolutivos, es usado en forma indistinta para describir diferentes técnicas de computación evolutiva.

Para las tareas de optimización de funciones de variables reales, la evolución estratégica ha emergido como una gran alternativa a diversos métodos de solución tradicionales.

La computación evolutiva plantea los problemas complejos de búsqueda y optimización bajo un enfoque evolutivo, basado en la teoría evolucionista. La computación evolutiva es una rama de la inteligencia artificial que involucra problemas de optimización combinatoria. Se inspira en los mecanismos de la evolución biológica.

Durante los años 60 y 70, varias corrientes de investigación independientes comenzaron a formar lo que ahora se conoce como computación evolutiva:

- Programación Evolucionaria o EP (Evolutionary Programming).
- Estrategias Evolutivas o ES (Evolution Strategies).
- Algoritmos Genéticos o GA (Genetic Algorithms).

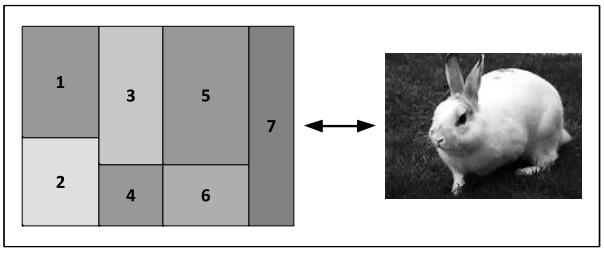
La Programación Evolutiva nació en la década de 1960 y su creador fue Lawrence J. Fogel. Este desarrollo comenzó como un esfuerzo encaminado a crear inteligencia artificial basada en la evolución de máquinas de estado finitas.

Las Estrategias Evolutivas fueron propuestas por Ingo Rechenberg y Hans Paul Schwefel en la década de 1970. Su principal objetivo era el de optimizar de parámetros.

Cada una de estas técnicas fue desarrollada de forma independiente, siendo la denominación computación evolutiva porque están basadas en un esquema general común ya que todos ellos poseen:

- Un conjunto de soluciones debidamente codificadas (los individuos) formando una población (ver la figura 2.11)
- Un procedimiento de transformación a la nueva población, desde la población anterior.
- Una función evaluadora del mérito de las soluciones que forman parte de la población.
- Un mecanismo de selección de individuos en función de su adaptación al medio.

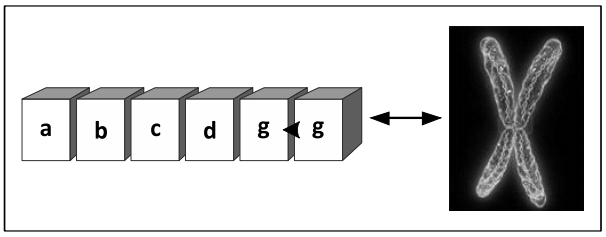
Figura 2.11. Solución y el individuo



Fuente: Elaboración propia.

El evolucionismo como un proceso de optimización, está ligado al genotipo de un individuo (ver figura 2.12), porque éste designa la constitución genética de un individuo; es decir es la información contenida en los alelos de los genes del individuo. Fenotipo corresponde a las características físicas o rasgos del individuo (altura, color de los ojos, etc.)

Figura 2.11. Codificación y cromosoma.

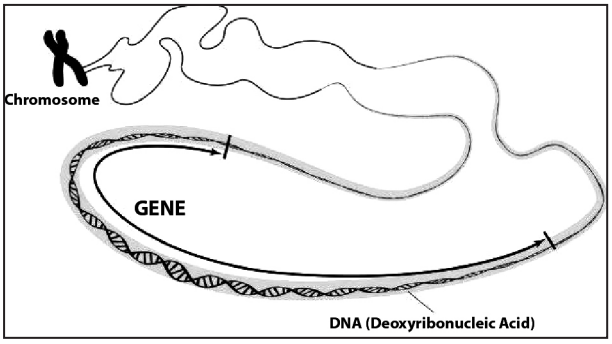


Fuente: Elaboración propia.

Los genes son arreglados uno después de otro en una estructura denominada cromosoma (ver la figura 2.12).

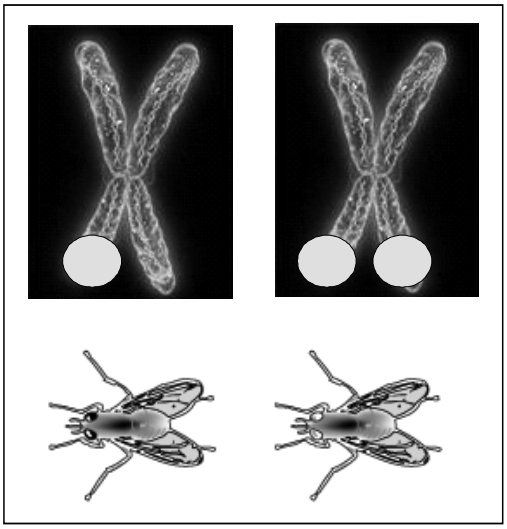
Los rasgos fenotípicos no son necesariamente consecuencia de la carga genética del individuo (ver figura 2.13 alelos y color de los ojos), sino que su conducta de éste frente al mundo exterior está condicionada por el ambiente: la epigénesis, y por el aprendizaje del individuo y del grupo a lo largo de su existencia. La epigénesis predice que los órganos del embrión son formados de la nada, por medio de inducción por parte del ambiente.

Figura 2.12. Estructura del cromosoma



Fuente: National Human Genome Research Institute (www.genome.gov)

Figura 2.13. Carga genética



Fuente: Elaboración propia.

La epigénesis representa por tanto el proceso de "sintonización" final mediante el cual cada individuo se adapta de forma eficiente a su entorno a partir de las capacidades contenidas en su código genético. Los genes son parte de una red compleja de interacciones que se retroalimenta y, por ende, no actúan como identidades independientes.

La adaptabilidad del individuo, está condicionada a la forma como se expresa su carga genética, es decir su fenotipo; no existiendo una relación directa entre el código genético y el comportamiento desarrollado.

2.2.4. Algoritmos Genéticos

Los Algoritmos Genéticos fueron propuestos por John H. Holland en 1975 y su motivación inicial fue la de proponer un modelo general del proceso de adaptación. En el libro *Adaptation in Natural and Artificial Systems* en 1975, J. Holland acuña el término Algoritmo Genético, universalmente conocido en nuestros días como GA (Genetic Algorithms).

Un algoritmo genético, es un tipo de sistema basado en la evolución, y corresponde a una clase de algoritmo de programación estocástica adaptativo, que envuelve búsqueda y optimización. Holland creó un organismo electrónico, como una cadena de binarios o string de bits (denominado cromosoma), haciendo uso de los principios de la genética y la evolución de las especies, tales como la selección y la reproducción incluyendo cruce aleatorio (o crossover) y mutación (Raffo Lecca, E., Ruiz, E., 2005).

La influencia de Holland en el desarrollo de este tópico es muy importante; junto con Ingo Rechemberg (1973) y Hans-Paul Schwefel (1977), comparten las ideas de mutación y selección, formando el núcleo de la teoría de la evolución neo Darwiniana (Handbook of Metaheuristics, 2003).

David Goldberg un estudiante de Holland, aplica los conceptos de GA en su tesis doctoral sobre la de optimización del transporte de gas en 1985, y más tarde publica el libro *Genetic Algorithm in Search, Optimization, and Machine* Learning en 1989. El cual fue el detonante para el crecimiento explosivo de GA.

En 1990, GA entra a la escena de la Investigación de Operaciones (Handbook of Metaheuristics, 2003).

La resolución de un problema, es la búsqueda de una solución óptima en un gran espacio de soluciones. La naturaleza se enfrenta al mismo dilema en la búsqueda de la mejor adaptación de los individuos al medio. Los integrantes de una población compiten entre ellos en la búsqueda de la supervivencia. Aquellos miembros de la población capaces de adaptarse mejor al medio que les rodean, tendrán mayor oportunidad de sobrevivir. Por otro lado los integrantes de la población con menos capacidades, tendrán una oportunidad menor de sobrevivir.

El fenotipo del individuo que ha logrado el éxito o fracaso, corresponde a la información contenida en su carga genética. Esta información se encuentra codificada en los genes (fragmento del ADN) en una determinada localización de un cromosoma específico. Los cromosomas se encuentran en el interior del núcleo.

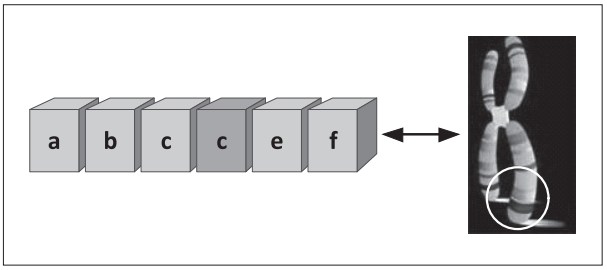
En la reproducción, los ancestros transmiten a su descendencia parte de su carga genética. En este cruce los individuos descendientes poseerán características del fenotipo de sus ancestros. Debido a que los individuos mejor dotados poseerán mayor descendencia, las sucesivas generaciones disfrutarán de la combinación de las buenas características de generaciones pasadas, lo que se traducirá en una mejor adaptación al medio.

Las mutaciones ocasionadas en el material genético, son otra fuente de variabilidad genética; produciendo cambios en la población, diferentes a las heredadas de las generaciones anteriores.

Los individuos mejor adaptados se reproducen, sobreviven y se reproducen en mayor medida dando lugar a lo que se ha denominado "la selección natural".

La Genética, plantea que los cromosomas contienen alrededor de 80.000 genes, los responsables de la herencia (ver la figura 2.14).

Figura 2.14. Genes en un cromosoma.



Fuente: Elaboración propia.

Los GA según D. Goldberg son algoritmos de búsqueda basados en los mecanismos naturales de selección y genética natural. Ellos combinan en la supervivencia estructuras del mejor ajuste con cambios de estructuras de información aleatoria (Goldberg, 1989).

GA difiere de la optimización tradicional por generar una población de puntos (no un punto), y seleccionar la siguiente población utilizando cambios aleatorios (en vez de conseguir un nuevo punto, desde procedimientos determinísticos).

Al asumir un espacio discreto χ y una función

$$f: \chi \to \mathbb{R}$$

El problema general es encontrar

$$\min_{x \in \chi} f$$

Aquí *X* es un vector conocido como el vector de variables decisionales, y *f* es la función objetivo. Se asume aquí que el problema es uno de minimización, siendo obvias las modificaciones a un problema de maximización. Estos problemas son conocidos como problemas discretos o de optimización Combinatoria o COP (*Combinatorial Optimization Problem*).

Una de las diferencias de GA, es permitir la separación de la representación del problema de las actuales variables del cual fue originalmente desarrollado. En otras palabras, se distingue el genotipo y la representación codificada de las variables desde el fenotipo. El vector *X* es representado por una cadena *s* de longitud *l,* realizado desde un alfabeto *A* usando el mapeo:

$$c: \mathcal{A}^l \to X$$

En la práctica, se necesita usar un espacio de búsqueda que cumple la condición:

$$\mathcal{S} \subseteq \mathcal{A}^l$$

La imagen de A^l bajo *c* puede representar una solución inválida al problema original. La cadena o string de longitud *l* depende de las dimensiones de ambos $\mathcal{X}$ y A , y los elementos de la cadena corresponden a los genes, siendo los alelos los valores de los genes.

Este es el mapeo genotipo-fenotipo, y el problema es encontrar la optimización:

$$\min_{s \in \mathcal{S}} g(s)$$

Donde la función es

$$g(s) = f(c(s))$$

Con *c* inyectiva, desde el punto que tiene una inversa.

En los libros de Holland y Goldberg, se plantea la representación de las variables como un string binario, es decir A={0,1}.

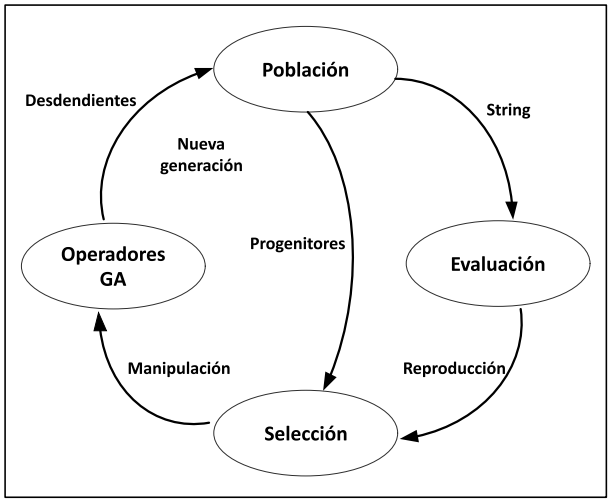
La motivación original de GA, fue la analogía biológica. En GA, una población de string es referido como cromosomas. La recombinación de las cadenas se realiza haciendo uso de simples analogías de cruce genético (crossover) y mutación, donde la búsqueda es guiada por los resultados de la evaluación de la función objetivo para cada string de la población. Basado en esta evaluación, se tienen los mejores ajustes o mejores soluciones, las que son identificadas por ofrecer mejores oportunidades.

Para aplicar GA a un problema, el primer paso consiste en codificar el cromosoma artificial. Estos pueden ser cadenas de unos y ceros, lista de parámetros, etc. Luego existe un procedimiento para discriminar las soluciones buenas de las malas. Es la función de evaluación o función de *fitness*, que es usada por GA para guiar la evolución de las nuevas generaciones. En este momento se está con las condiciones de evolucionar soluciones para el problema.

Los operadores genéticos, son utilizados para procesar iterativamente la población, creando una secuencia de poblaciones (ver la figura 2.15).

GA es un método para resolver problemas de optimización que está basado en la selección natural, el proceso que dirige la evolución biológica. El GA repetidamente modifica soluciones de poblaciones individuales. En cada paso, el GA selecciona aleatoriamente individuos desde la actual población produciendo los hijos de la nueva generación. Sobre sucesivas generaciones, la población se dirige a la solución óptima.

Figura 2.15. Ciclo de un GA



Fuente: Elaboración propia.

Los algoritmos de búsqueda aleatoria, han incrementado su popularidad en el campo de la investigación, por ser un atajo a los esquemas de la enumeración exhaustiva y las búsquedas basadas en el cálculo.

GA hace uso de tres reglas principales en cada iteración, para crear la nueva generación a partir de la actual población:

- Reglas de selección, para seleccionar los parientes que contribuyen a la población en la siguiente generación.
- Reglas de cruce que combinan dos parientes para formar el hijo de la siguiente generación.
- Reglas de mutación, que aplicar aleatoriamente cambios a los individuos parientes para formar hijos.

A modo de ejemplo considere una población con cuatro cadenas: 01001, 11010, 01001 y 10010. Evaluando las cadenas se encuentra los ajustes dados en la tabla 2.1. Los bits de las cadenas son generados aleatoriamente.

Tabla 2.1. Un problema simple de GA con ajustes de valores

Número	Cadena			% del total
1	01001	9	63	5
2	11010	26	624	51
3	10001	17	255	21
4	10010	18	288	23
Total		70	1230	100.00

Fuente: Elaboración propia.

La reproducción es el proceso en el cual las cadenas individuales son copiadas de acuerdo a los valores de la función objetivo $f(x) = x^2 - 2x$ (función de ajuste o *fitnees function*, según los biólogos).

La cadena 2 o el string 2, tiene un *f* de 624 y representa el 51% del total de los ajustes. Representado en una ruleta, este ocupa el 51% del área total del círculo. Se observa que su ajuste es del mayor valor. Ver la figura 2.16.

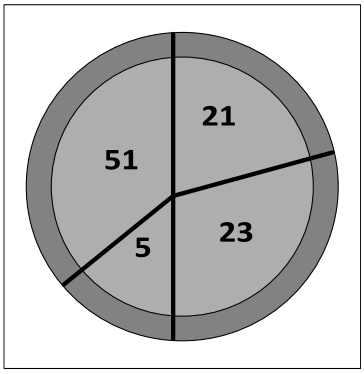
Una nueva generación empieza con la reproducción, se observa que uno de ellos tiene más posibilidad de ser elegido. Una vez que el string es seleccionado se realiza una copia del mismo.

Goldberg (1989) presenta un simple mecanismo de cruce:

Para cada cadena, seleccionar la pareja string a intercambiar.

Dentro de cada cadena, generar un número aleatorio entre 1 y la longitud de la cadena-1. El cruce se realiza desde el Indice+1 hasta la longitud de la cadena.

Figura 2.16. Un simple mecanismo de reproducción.



Fuente: Elaboración propia

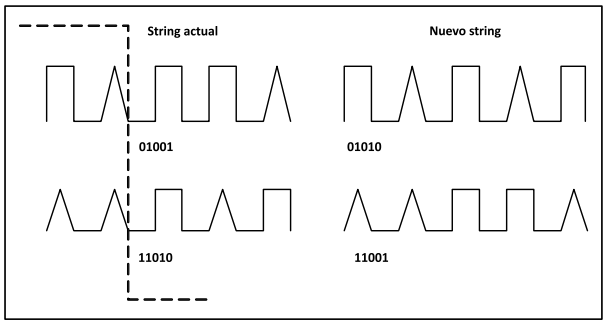
Tabla 2.2. Mecanismo de cruce para las cadenas.

Número	Cadena	Pareja	Índice	Nueva cadena	
1	01001	2	2	01010	10
2	11010	1	2	11001	25
3	10001	4	4	10000	16
4	10010	3	4	10011	19

Fuente: Elaboración propia.

En la tabla 2.2, se observa que la pareja de cadenas 1 y 2 intercambian sus elementos después del segundo valor. Ver la figura 2.17.

Figura 2.17. Un simple mecanismo de cruce



Fuente: Elaboración propia.

La mutación altera arbitrariamente uno o más genes del cromosoma seleccionado, mediante un cambio aleatorio con una probabilidad igual a la tasa de mutación. La intuición que se encuentra detrás del operador de mutación, es la introducción de algunas variabilidades extras en la población (Michalewicz, 1996).

Sea una población compuesta de 50 cromosomas (tamaño de la población) y cada cromosoma contiene 20 genes (20 bits), entonces se tienen 50x20= 1000 bits. Si la probabilidad de mutación es , en promedio se esperan por cada generación:

p_m x1000=0.01x1000=10

Mutaciones

Según la lógica de la programación de computadoras: por cada bit se genera un número aleatorio *R*, y si cumple que $R < p_m$ entonces el bit es mutado.

Según Michalewicz (1996), existen cinco componentes a cualquier GA(o cualquier programa evolutivo), tal como se presentan en la figura 2.18.

2.2.5. Algoritmos genéticos y distribución de plantas

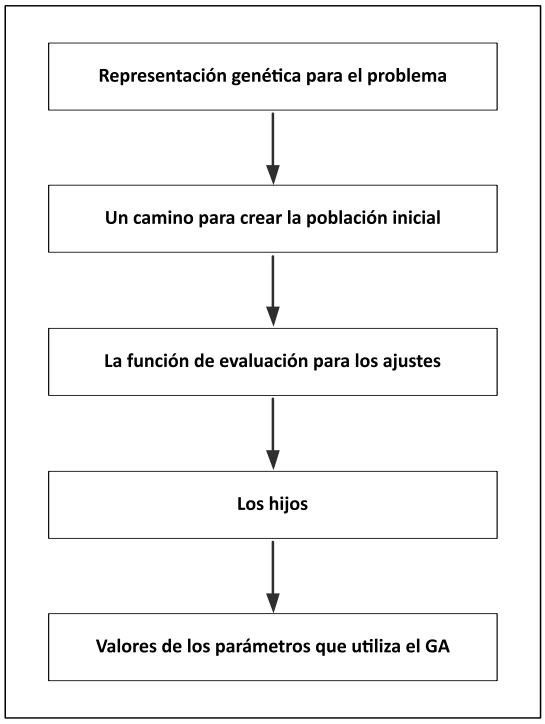
El problema del layout de planta o *FLP* (del inglés *Facility Layout Problem*) es, por lo general, muy difícil de resolver. El mismo ha sido clasificado como *NP-hard*, es decir, no se ha encontrado un algoritmo polinómico para resolverlo, por lo que el tiempo para encontrar una solución crece exponencialmente con respecto al tamaño del problema [Papadimitriou, 1994].

El tiempo y los recursos de computación requeridos para los problemas *NP-hard* en las aplicaciones prácticas vienen a ser prohibitivo. Siendo los métodos metaheurísticos la única alternativa viable (Arostegui y otros, 2006).

Desde la segunda mitad de la centuria 20 a la presente, diferentes metodologías han sido introducidas para la solución del FLP. Técnicas computarizadas para diseño de un nuevo layout o mejoramiento de las existentes se iniciaron con CRAFT [Armour y Buffa, 1963], CORELAP [Sepponen, 1969], COFAD [Tompkins y Redd Jr., 1976], ALDEP [Seehof y Evans, 1967] y PLANET [Konz, 1985]. La tendencia actual es el uso de propósitos multi-objetivos con procedimientos basados en Metaheurísticas, destacando entre ellos GA, TS, SA, ACO, PSO, sistemas híbridos y aplicación de la realidad virtual [Kundu y Dan, 2010].

De los métodos de la metaheurística, destacan TS (Tabu Search), SA (Simulated Annealing) y GA (General Algorithms), debido a que en años recientes han incrementado su popularidad, por la diversa literatura publicada en torno a las aplicaciones a FLP [Arostegui y otros, 2006].

Figura 2.18. Componentes de un GA.



Fuente: Elaboración propia (basado en Michalewicz, 1996).

El estado del arte en la aplicación de GA al importante problema FLP, puede ser visto como la tendencia emergente en nuevos diseños de objetivos y metodologías, que está direccionando la Optimización Combinatoria o COP [Kundu y Dan, 2010].

El mecanismo de búsqueda de GA está basado en la selección natural y la genética natural. Donde su amplia difusión es debido a su robustez (Goldberg, 1989).

De acuerdo al estudio de aplicaciones de GA, realizado por Kundu y Dan (2010) para la optimización de diversas formulaciones para FLP, se usan seis principales objetivos:

A. Minimización del manejo o costo total del manejo de materiales.
B. Minimización del tamaño o maximización del área utilizada.
C. Generación de un layout flexible.
D. Minimización del costo de manejo de material del movimiento de celdas.
E. Minimización de la distancia rectilínea total del material.
F. Optimización de la estructura.

Tabla 2.3. Aplicaciones usando GA

Referencias	Técnicas de modelación	Objetivos					
		A	B	C	D	E	F
Delmaire	LP	x					
Diego-Mas	Slicing Tree	x					
Dunker	DP	x					
Eklund	MIP	x		x			
El-Baz	QAP	x					
Gau, Meller	MIP	x					
Hicks	Clustering					x	
Hu & Wang	QAP	x	x				
Kochhar	MIP	x		x			
Kulkani	QAP	x					
Lee & Lee	QAP	x	x				
Liu & Li	MIP	x					
Longo	QAP	x					
Osman	QAP	x					
Rajasekharan	MIP	x					
Ramkumar	QAP	x					
Wu & Appleton	QAP	x					x
Wu	MIP	x			x		

Fuente: Kundu y Dan, 2010.

La tabla 2.3 es el resultado de éste estudio, donde se desprende que es significante la optimización de modelos QAP (del inglés *Quadratic Assignment Problem*), seguido de modelos de programación entera mixto o MIP (*Mixed Integer Problem*). El objetivo más utilizado es del tipo A.

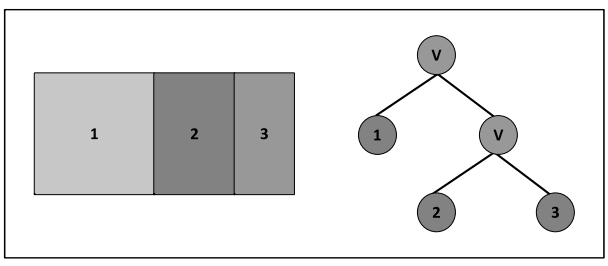
2.2.6. Representación de bloques layout

El incremento de la complejidad en los circuitos integrados motivado por el desarrollo de las metodologías para el diseño automatizado VLSI(Very Large Scale Integrated), ha generado la aparición de una nueva clase de metodologías, contándose entre ellos a Zibert (1974), Lauther (1980), Otten (1982) y otros autores (Stockmeyer, 1983). Estas metodologías describen la posición relativa de las piezas indivisibles, conocidas como celdas en un chip o áreas en un FLP; y en general el diseño *floorplan*, acepción más generalizada cuando se aplica a problemas del mundo real cuya tarea es encontrar propiedades de representación para la solución de los candidatos (áreas o piezas), en problemas COP 2D, tales como *Packing Problems*, QAP, entre otros.

Un importante problema en la aplicación de la Metaheurística al FLP es desarrollar la codificación de los candidatos en el bloque layout. La formulación de la estructura de árbol de corte o STS (del inglés *Slicing Tree Structure*) definida por Ralph Otten (1982), es una codificación continua del layout en una estructura árbol o *tree*. Ver figura 2.19.

La jerarquía de cortes creada por recursividad, divide un área rectangular mediante cortes en direcciones horizontales o verticales, siendo la implementación más simple la binaria que representa patrones de guillotina o de corte. Esta formulación permite a los bloques layout ser localizados en zonas divididas, y toma ventaja de la forma y la orientación flexible, tal como necesita el tratamiento de los bloques layout. Un *slicing tree* es el equivalente a la expresión polaca RPN(Reverse Polish Norm) .

Figura 2.19. Operador vertical en una SST.



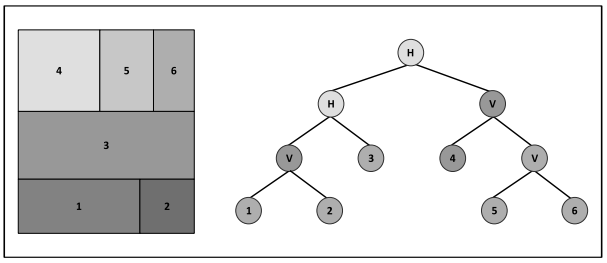
Fuente: Elaboración propia.

Las hojas nodos en una representación STS representan los módulos. Cada nodo interno es rotulado por un operador *V* o *H*, dependiendo de la línea de corte respectivamente. En la figura 2.18 las líneas de corte son verticales. La expresión polaca para un SST, es obtenida usando el recorrido *postorder*.

En la figura 2.20 se presenta el arreglo de bloques layout conocido como la estructura SST y el árbol de corte para un layout compuesto de seis departamentos.

A partir de Schnecke y Vornberger (1998), Otten implementa funciones de formas o *Shape functions*, dando flexibilidad a las celdas para que jerárquicamente puedan ser comprimidas en un conjunto de sub-celdas.

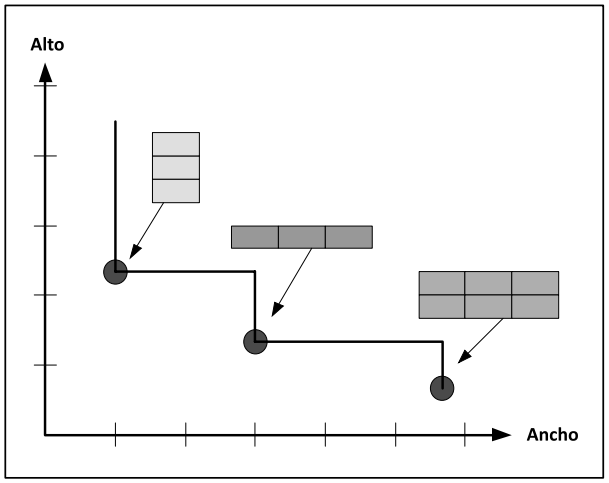
Figura 2.20. Estructura SST y Slicing tree



Fuente: Elaboración propia.

En la figura 2.21 se presentan tres bloques, conteniendo los dos primeros 3 celdas y el último seis celdas; constituyendo metas bloques.

Figura 2.21. Estructura SST y Slicing tree.



Fuente: Elaboración propia.

Tam y Chan (1998), representan el layout floorplan como una estructura slicing o SST, tal como se visualiza en la figura 2.19, donde cada hoja del árbol está asignada a un número entero correspondiente a su identificador (1, 2, etc.). Cada nodo interno del árbol acomoda a un operador el que especifica el camino de la porción rectangular a cortar (V de vertical y H de horizontal). Este esquema de código representa el layout como una cadena de caracteres o string.

Al contener el esquema de código tres partes o sub-cromosomas (Al-Hakim, 2000): el árbol, el nodo interno y las salidas externas, se plantea tres operaciones para encontrar las diferentes soluciones:

- Cambio de las estructura del árbol.
- Cambio del símbolo del nodo interno.
- Cambio del valor entero de las salidas.

Tam y Chan (1998) usan la representación bit-string para codificar las partes del cromosoma. Al usar "0" y "1" para denotar los nodos internos y las salidas respectivamente, se encuentra que el primer elemento es "0" y los dos últimos elementos son "1"; el total de 0s es N - 1, siendo N el número de facilidades (departamentos) que corresponde al número de nodos de salidas. Además en cualquier posición P el número de "0" que aparecen antes es mayor que el número de "1"antes de P**.**

Desde la figura 2.20, se tiene el bit-string . Como el primer elemento y los dos últimos son conocidos son removidos de la estructura. De la misma manera el tercer último elemento también es removido, resultando la representación .

Al-Hakim (2000) plantea que esta representación restringe la aplicación de los dos clásicos operadores en GA: cruce y mutación, toda vez que existe una alta posibilidad que estos operadores produzcan *offspring* no factibles, dicho de otro modo una generación de individuos nuevos no factibles. Al-Hakin plantea nuevos operadores: transplante, cruce diagonal y clonación.

CONCLUSIONES

La revisión del estado del arte proporciona el camino que están siguiendo los investigadores en la aplicación de los algoritmos genéticos al problema de la distribución de planta.

REFERENCIAS

[1] Al-Hakim, L. (2000). On Solving Facility Layout Problem Using Genetic Algorithms. International Journal of Production Research, Vol 38, N0 11 pág. 2573-2582.

[2] Armour, G., Buffa, E. (1963). A Heuristic Algorithm and Simulation approach to Relative Allocation of Facilities. Management Sciences, 9 pág. 294-309.

[3] Arostegui, M., Kadipasaoglu, S., Khumawala, B. (2006). An empirical comparison of Tabu Search, Simulated Annealing, and Genetic Algorithms for facilities location problems. International Journal Production Economics 103(2006) 742-754.

[4] Bernal T., César (2006), Metodología de la Investigación para la Administración, Economía, Humanidades y Ciencias Sociales. México D.F., México: Pearson Educación.

[5] Raffo Lecca, E., Ruiz, E. (2005). Optimización por Computación Evolucionaria, Industrial Data, Vol. 8, No. 2, Pág. 61-68, 2005.

[6] Glover, F., Kochenberg, G. (2003). Handbook of Metaheuristics. Massachusetts, USA: Kluwer Academic Publishers.

[7] Goldberg, David (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. New York, USA: Addison-Wesley.

[8] Francis, R., McGinnis, L., White, J. (1992). Facility Layout and Location: An Analitical Approach. New Jersey, USA: Prentice-Hall, Inc., Englewood Cliffs, second edition.

[9] Hernández Sampieri, R., Fernández, C., Baptista, P. (2014) Metodología de la Investigación. México D.F., México: Editorial Mc Graw-Hill.

[10] Konz, S. (1985). Facility Design. John Wiley & Sons.

[11] Kundu, A., Dan, P. (2010). The Scope of Genetic Algorithms in Dealing with Facility Layout Problem. South African Journal of Industrial Engineering Nov. 2010 Vol. 21(2): 39-49.

[12] Michalewicz, Z. (1996). Genetic Algorithms + Data Structures = Evolution Programs. New York, USA: Springer-Verlag series Artificial Intelligence.

[13] Mitchell, Melanie. (1996). An Introduction to Genetic Algorithms". MIT Press.

[14] Muther, R. (1961). Systematic Layout Planning. Boston, USA: Industrial Education Institute.

[15] Muther, R. (1997). Distribución en Planta. Barcelona, España: Editorial Hispano Europea S.A.

[16] Nadler, G. (1961). Work Design: A Systems Concept. USA: Richard D. Irwin, Inc., Homewood, IL.

[17] Papadimitriou, C. (1994). Computational Complexity. USA: Addison Wesley.

[18] Phillips, D., Ravindran, A., Solberg, J. (1976). Operations Research: Principles and Practice, John Wiley & Sons, Inc.

[19] Otten, Ralph (1982). Automatic floorplan design. Proceeding of the 19th Design Automation Conference, pág. 261-267.

[20] Seehof, J., Evans, W. (1967). Automated Layout Design Program. The Journal of Industrial Engineering, 18(12), pág. 690-695.

[21] Sepponen, R. (1969). CORELAP8 User Manual. Department of Industrial Engineering, Northeastern University, Boston.

[22] Stockmeyer, L. (1983). Optimal Orientations of Cell in Sclicing Floorplan Designs. Information and Control 57, pág. 91-101.

[23] Tam, K., Chan, S. (1998). Solving Facility Layout Problems with Geometric Constraints using Parallel Genetic Algorithms: Experimentación and Findings. International Journal of Production Research, Vol. 36, pág. 3253-3272.

[24] The MathWorks (2008) Manual de Algoritmos Genéticos MATLAB. USA.

[25] Tompkins, J., Reed Jr., R. (1976). An Applied model for the Facilities Design Problem. International Journal of Production Research, 14(5), pág. 597-602.

[26] Tompkins, J. (2007). Distribución de las plantas. México D.F., México: Editorial Limusa, S.A., de C.V. GRUPO NORIEGA EDITORES, Gavriel Salvendy, Manual de Ingeniería Industrial Volumen II, Capítulo 10.2.

[27] Velasco, J. (2010). Organización de la Producción, Distribuciones en Planta y Mejora de los métodos y los tiempos. Teoría y Práctica. Madrid, España: Ediciones Pirámide (Grupo Anaya, S.A.), segunda edición.