Aplicación de la programación entera binaria para resolver el problema simple de balanceo de linea de ensamble: un caso de estudio

Binary integer programming application to solve the simple assembly line balancing problem: a study case.

Daniel Felipe Escobar Alvarán, Julián Alberto Garcés Hincapié, Jorge Hernán Restrepo Correa
Ingeniera Industrial, Universidad Tecnológica de Pereira, Pereira, Colombia
descobaralvaran@hotmail.com
juliangarceshincapie@hotmail.com
jhrestrepoco@utp.edu.co

***Resumen*— Este documento presenta en forma reducida la aplicación de la programación lineal para solucionar un problema simple de balanceo de línea de ensamble. El tipo de programación utilizada es la programación entera binaria, se muestran los pasos de la aplicación de este método para resolver problemas de tipo combinatorio.**

***Palabras clave*— Estación, Tarea, Tiempo de ciclo, Algoritmo, Programación entera binaria.**

***Abstract*— This paper shows in a reduced form the application of the lineal programming to solve a simple assembly line balancing problem. The type of programming, used is the binary integer programming. The steps for the application of this method to solve combinatorial problems are shown.**

***Key Word* — Workstation, Task, Cycle time, Algorithm, Binary integer programming**

I. INTRODUCCIÓN

El problema de las líneas de ensamble ha sido estudiado por diferentes investigadores desde diferentes enfoques, tratando de dar solución a este problema utilizando algoritmos exactos y algoritmos heurísticos. La noción de línea de ensamble existe desde hace varios siglos, cuando se tenían líneas incipientes y rudimentarias para construir embarcaciones en astilleros en Venecia; pero solo fue hasta Henry Ford que se estudiaron las líneas de ensamble de manera técnica y científica, desde ese momento empieza la investigación y desarrollo sobre el equilibrado de las líneas de ensamble. La optimización de las líneas de ensamble ha sido el objetivo de las investigaciones en las cuales se han empleado diferentes técnicas: heurísticas, meta-heurísticas, algoritmos exactos, algoritmos genéticos. Entre otros. [1]

La aparición de los modelos matemáticos lineales con fines aplicados a la optimización de recursos se remonta a la Segunda Guerra Mundial, en donde era utilizada con fines de planificar los gastos, reduciendo así costes y maximizar las pérdidas al enemigo. Esta práctica se mantuvo en secreto hasta 1947. Al terminar la guerra la industria acogió estos modelos con el fin de minimizar costos, maximizar el uso del tiempo, entre otros usos como el balanceo de líneas.

El problema de balanceo de líneas de ensamble, es uno de los más comunes en las fábricas y empresas industriales, en términos generales trata de optimizar los recursos de la línea de ensamble, ya sea minimizando estaciones de trabajo, o minimizando el tiempo de ciclo, es decir, el problema de balanceo de línea de ensamble trata de asignar las tareas en una secuencia ordenada de las estaciones, satisfaciendo las relaciones de precedencia y optimizando una función objetivo. [1]

En este trabajo se resolverá concretamente un pequeño problema tipo SALBP-1 (será descrito posteriormente), utilizando la programación dinámica.

II. CLASES DE PROBLEMAS DE BALANCEO DE LINEA DE ENSAMBLE (Tomado de [1])

Consiste en distribuir las tareas necesarias para ensamblar un producto a través del conjunto de estaciones que conforman la línea de ensamble, esta distribución de las tareas en las estaciones de trabajo se hace siguiendo un objetivo, puede ser maximizar la eficiencia de la línea, o minimizar el tiempo ocioso o minimizar el número de estaciones requeridas en la línea de ensamble. Un problema de balanceo de línea está compuesto por una función objetivo y un conjunto de restricciones.

Una solución factible de un problema de balanceo de línea de ensamble debe cumplir con las siguientes condiciones:

- Cada tarea se debe asignar exactamente a una estación
- Se debe cumplir por completo con las relaciones de precedencia.

Fecha de Recepción: 12 de Mayo de 2011
Fecha de Aceptación: 18 de Diciembre 2011

- La suma de los tiempos de las tareas de cada estación no deben exceder el tiempo de ciclo, para todas las estaciones.

El problema de balanceo de línea de ensamble (ALBP) se divide en dos categorías, los SALBPs, que son problemas simples de balanceo de línea, en los que se consideran pocas variables de entrada desconocidas para reducir la complejidad del mismo; y los GALBPs, problemas generales de balanceo de línea de ensamble, que estudian casos más reales y complejos que se presentan en la industria.

A. Problema general (GALBP)

Según el documento de Capacho y Pastor [3], los problemas generales de balanceo de líneas de ensamble, consideran los problemas que no son SALBP, es decir, problemas más complejos, cuyas características se asemejan a un problema real de balanceo de línea.

Se distinguen cuatro casos de GALBP:

UALBP: U-line assembly line balancing problem – problema de equilibrado de líneas tipo U. Los UABLP están caracterizados de manera similar a los problemas SALBP pero consideran una línea tipo U en lugar de una serial. Las líneas tipo U se consideran líneas más flexibles que las líneas tipo serial, según Scholl y Becker [9], en los SALBP únicamente se pueden asignar aquellas tareas cuyos predecesores han sido asignados. Las estaciones pueden ser colocadas de tal manera que, durante el mismo tiempo de ciclo, se puedan manejar a la vez dos piezas en diferentes posiciones de la línea. Esto implica que hay un mayor número de posibilidades de asignar las tareas a las estaciones, lo que resulta, en algunos casos que el problema se pueda resolver de manera más eficiente que cuando se tiene un línea simple. De manera similar a los problemas simples SALBP, se distinguen los problemas UABLP-1, UABLP-2 y UABLP-E, en donde se busca minimizar el número de estaciones, minimizar el tiempo de ciclo y maximizar la eficiencia de la línea U, respectivamente.

MALBP: mixed-model assembly line balancing problem Problema de equilibrado de líneas de modelos mixtos. Este tipo de problemas se presentan cuando se consideran varios modelos de un mismo producto y, por lo tanto, se tiene un conjunto de tareas básicas que se realizan en todos los modelos sin considerar tiempos de Setup. En este caso, también se tiene el problema de secuenciación de los diferentes modelos así como el problema de determinar el tamaño de los lotes de cada modelo; la secuenciación puede ser importante dado que los tiempos de tareas entre modelos pueden variar significativamente. También se tienen las versiones MALPB-1, MALBP-2 y MALBP-E. [3]

RALBP: robotic assembly line balancing problem – problema de equilibrado de líneas robotizadas. En este tipo de problemas se considera tanto la asignación de las tareas como la asignación de un grupo de robots a las estaciones de trabajo, con la finalidad de optimizar la realización de las tareas en la línea.

MOALBP: multi-objective assembly line balancing problem – problema de equilibrado de líneas con objetivos múltiples. En este tipo de problemas se consideran varios objetivos simultáneamente como por ejemplo: minimizar el número de estaciones, el coste total de montaje o el número de buffers; maximizar la eficiencia de línea, etc. De acuerdo con Capacho y Pastor [3] la mayoría de los problemas de equilibrado de líneas consideran múltiples objetivos.

Los problemas anteriores, el SALBP y el GALBP se pueden subdividir, teniendo en cuenta:
El tipo de producto que se procesa en la línea: modelo simple (SM) y modelo mixto (múltiple) (MM).
La variabilidad del tiempo de duración de las tareas: determinístico (D) y estocástico (S).

B. PROBLEMA SIMPLE (SALBP)

Los SALBP contienen los problemas de balanceo más simples y se caracterizan por: consideran líneas simples, sólo se consideran restricciones de precedencia, se asume que las tareas son indivisibles, los tiempos de proceso de las tareas son considerados independientes de la estación y del orden de proceso, los tiempos de proceso de las tareas son determinísticos y conocidos a priori, así como todos los parámetros de entrada, la línea es sincrónica, se tiene un tiempo de ciclo (o un número de estaciones) fijo, la arquitectura de la línea es serial con todas las estaciones igualmente equipadas para realizar cualquiera de las tareas y la tasa de entrada de las piezas a la línea es fija.

Se distinguen cuatro casos de SALBP:

1. SALBP-1: consiste en asignar un conjunto de tareas a las estaciones de tal forma que se minimice el número de estaciones, dado un tiempo de ciclo (o tasa de producción). Este caso se presenta habitualmente cuando un nuevo sistema de montaje va a ser instalado y la demanda externa puede ser estimada.

2. SALBP-2: Este problema busca lo contrario del problema anterior, es decir, se busca minimizar el tiempo de ciclo (o maximizar la tasa de producción), dado un
Número de estaciones fijo. Se considera que la línea de montaje ya existe.

3. SALBP-E: maximiza la eficiencia E de la línea, es decir, minimiza el producto de m (número de estaciones) por c (tiempo de ciclo).
4. SALBP-F: consiste en determinar si existe alguna solución factible para la combinación de un número m de estaciones y un

tiempo de ciclo c; es decir, se quiere conocer si la línea puede operar con m estaciones y un tiempo de ciclo c dados. No se busca minimizar ni maximizar ningún valor.

El siguiente cuadro muestra en resumen las características de los problemas SALBP`s.

Versión	Tiempo de Ciclo	Número de Estaciones
SALBP-F	Fijo	Fijo
SALBP-1	Fijo	Mínimo
SALBP-2	Mínimo	Fijo
SALBP-E	Mínimo	Mínimo

Tabla 1. Versiones del SALBP

III. PROGRAMACIÓN ENTERA BINARIA

La programación entera binaria es un método perteneciente a la programación lineal, por lo que su base es un algoritmo matemático que tiene como finalidad resolver un problema indeterminado formulado a través de ecuaciones lineales, optimizando así una función objetivo también lineal que generalmente se refiere a costo o a tiempo.

La programación binaria se utiliza en problemas de asignación o de toma de decisiones enfocadas a hacer o no una tarea, entre sus campo de aplicación más comunes se encuentra el despacho de envíos, el diseño de redes, la elección de un sitio, el diseño de redes, la ubicación del personal y la programación de actividades, que es la aplicación objeto de estudio en este artículo.

IV. METODOLOGIA DE SOLUCION DE UN PEQUEÑO PROBLEMA SIMPLE DE BALANCEO DE LINEA DE ENSAMBLE

A continuación se muestra la metodología de solución utilizada para resolver el problema simple de balanceo de línea de ensamble, en dicho problema se conoce el tiempo de ciclo y se quiere encontrar el mínimo de estaciones para realizar las tareas de la línea.

A. Problema

Se tiene una línea de ensamble, la cual, sus características permiten abordar el problema de equilibrado como un problema SALBP. Se conoce que el tiempo de ciclo es de 100 segundos, se tienen los tiempos estándares de las tareas en la siguiente tabla:

Tarea	tiempo (s)	Predecesor in
A	40	-
B	75	A
C	50	A
D	35	C
E	80	B,D

Tabla 2. Tiempos de las tareas

El siguiente diagrama explica mejor el problema:

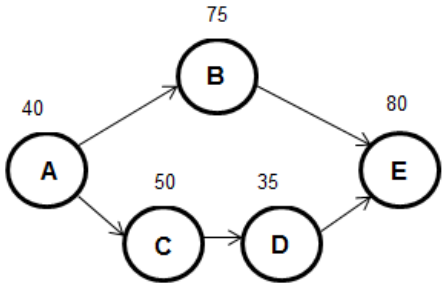

Figura 1. Gráfico de precedencias.

Figura 1. Gráfico de precedencias.

El objetivo en este problema es minimizar el número de máquinas a utilizar, partiendo del supuesto de que se necesita una máquina por cada tarea.

B. Solución

Identificamos el problema como un SALBP, tipo 1, ya que el tiempo de ciclo es dado, (100 segundos), se va a emplear la programación entera binaria para encontrar el mínimo número de máquinas o estaciones de la línea.

Se plantea la función objetivo de acuerdo al siguiente criterio.

$$Min\ Z = \sum_{i=1}^{N}\sum_{k=1}^{k} C_{ik}X_{ik}$$

Donde C_i = coeficiente de costo

Donde C_i = coeficiente de costoX_{ik} = 1, si la tarea i es asignada a la estación k; ó X_{ik} = 0, de lo contrario.

- Se plantean las restricciones de problema así:

Restricción de tiempo de ciclo: $\sum_{i=1}^{n} T_i\ X_{ik} \leq C$ para k= 1,2,…, k.

Restricción asignación unitaria: $\sum_{k=1}^{k} X_{ik} = 1$ para i =1,2,…..N

Restricción de precedencia: $X_{vb} \leq \sum_{j=1}^{b} X_{uj}$ para b= 1,2,…, k y (u, v) ϵ IP: Restricción de precedencia.

Teniendo en cuenta la metodología anterior se procede a resolver el problema propuesto:

Inicialización:

X_{ik} = Tarea i asignada a la estación k, donde i = a, b, c, d, e; Y k = 1, 2, 3, 4, 5.

Planteamos la función objetivo en donde añadimos pesos posicionales de 1, 2, 3, 4 y 5 a cada máquina respectivamente, con el fin de que la máquina o estación uno tenga prioridad sobre la 2, 3, 4 y 5, la 2 sobre la 3, 4 y 5; y así respectivamente, tal que si no es necesario el implemento de una estación de trabajo se ocupen las primeras estaciones prioritariamente.

Zmin=

$$X_{a1} + 2X_{a2} + 3X_{a3} + 4X_{a4} + 5X_{a5} + X_{b1} + 2X_{b2} + 3X_{b3} + 4X_{b4} + 5X_{b5} + X_{c1} + 2X_{c2} + 3X_{c3} + 4X_{c4} + 5X_{c5} + X_{d1} + 2X_{d2} + 3X_{d3} + 4X_{d4} + 5X_{d5} + X_{e1} + 2X_{e2} + 3X_{e3} + 4X_{e4} + 5X_{e5}$$

Ahora planteamos las restricciones empezando por la restricción del tiempo de ciclo:

(1) $40X_{a1} + 75\,X_{b1} + 50X_{c1} + 35X_{d1} + 80X_{e1} \leq 100$

(2) $40X_{a2} + 75\,X_{b2} + 50X_{c2} + 35X_{d2} + 80X_{e2} \leq 100$

(3) $40X_{a3} + 75\,X_{b3} + 50X_{c3} + 35X_{d3} + 80X_{e3} \leq 100$

(4) $40X_{a4} + 75\,X_{b4} + 50X_{c4} + 35X_{d4} + 80X_{e4} \leq 100$

(5) $40X_{a5} + 75\,X_{b5} + 50X_{c5} + 35X_{d5} + 80X_{e5} \leq 100$

A continuación se plantean las restricciones de asignación unitaria:

(6) $X_{a1} + X_{a2} + X_{a3} + X_{a4} + X_{a5} = 1$

(7) $X_{b1} + X_{b2} + X_{b3} + X_{b4} + X_{b5} = 1$

(8) $X_{c1} + X_{c2} + X_{c3} + X_{c4} + X_{c5} = 1$

(9) $X_{d1} + X_{d2} + X_{d3} + X_{d4} + X_{d5} = 1$

(10) $X_{e1} + X_{e2} + X_{e3} + X_{e4} + X_{e5} = 1$

Y por último planteamos las restricciones de precedencia, empezando con las correspondientes a la restricción de que el predecesor inmediato de b es a, así que a debe estar asignado para poder asignar b.

(11) $X_{b1} \leq X_{a1}$

(12) $X_{b2} \leq X_{a1} + X_{a2}$

(13) $X_{b3} \leq X_{a1} + X_{a2} + X_{a3}$

(14) $X_{b4} \leq X_{a1} + X_{a2} + X_{a3} + X_{a4}$

(15) $X_{b5} \leq X_{a1} + X_{a2} + X_{a3} + X_{a4} + X_{a5}$

El predecesor inmediato de c es a, cumpliéndose por lo tanto lo siguiente:

(16) $X_{c1} \leq X_{a1}$

(17) $X_{c2} \leq X_{a1} + X_{a2}$

(18) $X_{c3} \leq X_{a1} + X_{a2} + X_{a3}$

(19) $X_{c4} \leq X_{a1} + X_{a2} + X_{a3} + X_{a4}$

(20) $X_{c5} \leq X_{a1} + X_{a2} + X_{a3} + X_{a4} + X_{a5}$

El predecesor inmediato de d es c:

(21) $X_{d1} \leq X_{c1}$

(22) $X_{d2} \leq X_{c1} + X_{c2}$

(23) $X_{d3} \leq X_{c1} + X_{c2} + X_{c3}$

(24) $X_{d4} \leq X_{c1} + X_{c2} + X_{c3} + X_{c4}$

(25) $X_{d5} \leq X_{c1} + X_{c2} + X_{c3} + X_{c4} + X_{c5}$

Por último e tiene dos predecesores inmediatos, b y d, planteándose las restricciones así:

(26) $2X_{e1} \leq X_{b1} + X_{d1}$

(27) $2X_{e2} \leq X_{b1} + X_{d1} + X_{b2} + X_{d2}$

(28) $2X_{e3} \leq X_{b1} + X_{d1} + X_{b2} + X_{d2} + X_{b3} + X_{d3}$

(29) $2X_{e4} \leq X_{b1} + X_{d1} + X_{b2} + X_{d2} + X_{b3} + X_{d3} + X_{b4} + X_{d4}$

(30) $2X_{e4} \leq X_{b1} + X_{d1} + X_{b2} + X_{d2} + X_{b3} + X_{d3} + X_{b4} + X_{d4} + X_{b5} + X_{d5}$

Como se aprecia resulta un problema muy extenso con 30 restricciones, por lo cual para la solución se implementa un software libre denominado WinQSB, obteniendo como resultado lo siguiente:

Decision Variable	Solution Value	Unit Cost or Profit c(j)	Total Contribution	Reduced Cost
Xa1	1,00	1,00	1,00	0
Xa2	0	2,00	0	1,00
Xa3	0	3,00	0	2,00
Xa4	0	4,00	0	3,00
Xa5	0	5,00	0	4,00
Xb1	0	1,00	0	-3,50
Xb2	1,00	2,00	2,00	-1,00
Xb3	0	3,00	0	0
Xb4	0	4,00	0	1,00
Xb5	0	5,00	0	2,00
Xc1	1,00	1,00	1,00	-1,00
Xc2	0	2,00	0	0
Xc3	0	3,00	0	1,00
Xc4	0	4,00	0	2,00
Xc5	0	5,00	0	3,00
Xd1	0	1,00	0	-3,50
Xd2	0	2,00	0	-1,00
Xd3	1,00	3,00	3,00	0
Xd4	0	4,00	0	1,00
Xd5	0	5,00	0	2,00
Xe1	0	1,00	0	0
Xe2	0	2,00	0	-2,00
Xe3	0	3,00	0	-1,00
Xe4	1,00	4,00	4,00	0
Xe5	0	5,00	0	1,00
Goal	Value	(Min.) =	11,00	

Para el caso de este problema la función objetivo arroja un resultado de 11, sin embargo este valor para los fines del problema es irrelevante, ya que el fin real de este problema es la minimización del número de máquinas o estaciones de trabajo, arrojando de esta manera los siguientes resultados:

La tarea a se asigna a la máquina 1; la tarea b se asigna a la máquina 2; la tarea c se asigna a la máquina 1, la tarea d se asigna a la máquina 3 y la tarea e se asigna a la máquina 4. Como se puede observar no se asignó tarea alguna a la máquina 5, por ende esta es innecesaria, lo cual trae grandes ahorros ya que nos evitamos los costos y los tiempos de Setup que acarrea montar otra estación en la línea de ensamble. Se observa también que la tarea a y c se asignan a la misma estación minimizando así el número de estaciones a 4.

Para el problema propuesto, se tiene un mínimo de 4 estaciones de trabajo, ahora se va a calcular la eficiencia de la línea:

: Eficiencia de la Línea
: Número de estaciones utilizado
: Tiempo de ciclo
: Tarea i

VI. CONCLUSIONES.

La presente investigación se realizó con el fin de conocer la funcionalidad que posee el uso de la programación lineal, y más específicamente el uso de la programación entera binaria en la solución del problema de equilibrado de líneas de ensamble. Como se observó en el trabajo, el problema de balanceo de línea de ensamble abarca muchos temas, y en esta investigación solo se profundizo en la solución de problemas tipo SALBP-1, es decir, el más sencillo de los modelos entre los problemas de equilibrado de líneas.

El problema que se solucionó en el presente trabajo contiene 5 tareas elementales, es decir, es un problema muy pequeño, sin embargo sirve de modelo para aplicar la metodología de la programación entera binaria, ya que el principal objetivo de este trabajo era mostrar como es el procedimiento de solución de un problema simple de balanceo de línea de ensamble tipo SALBP-1.

Se aprecia que a pesar de ser un problema muy pequeño, resulta un número muy extenso de restricciones para la solución del problema, lo cual en un caso más grande y complejo puede resultar en un proceso largo y dispendioso, sin embargo gracias a los software como el WinQSB, se puede llegar a una solución clara y efectiva.
En consecuencia a medida que el problema crece en su número de operaciones, intentar una solución con programación binaria se vuelve complejo y poco práctico.

La programación entera binaria es una herramienta muy útil en cuanto al proceso de toma de decisiones de incluir o no cierto elemento y sus aplicaciones se extienden desde la ubicación de plantas de distribución en cierta ciudad, hasta en la decisión de un técnico de fútbol para incluir o no un jugador para el próximo juego, pasando claro está por problemas del tipo aquí tratados. Por tanto este tipo de programación es de vital importancia y puede ser de gran ayuda siempre que se piense en un problema de cualquier campo en donde se deba tomar una decisión sobre la inclusión o no de un elemento con el fin de optimizar los resultados deseados.

En la revisión de la bibliografía, se observó que la solución de los problemas de balanceo de línea de ensamble: SALBP y GALBP, todavía son objeto de estudio e investigación, cada día se desarrollan heurísticas y se aplican nuevos algoritmos para intentar obtener soluciones optimas con tiempos de cálculo reducidos; esto aún no ha sucedido, es decir, se han explorado diferentes formas de modelar y resolver el problema, y ningún autor menciona cual es la mejor forma de modelar, ni cuál es la mejor técnica de solución.
La solución a los problemas de balanceo de líneas de ensamble todavía es un tema abierto, en el cual se pueden realizar investigaciones y desarrollar técnicas más eficientes, ya que dependiendo del tipo de problema, se pueden utilizar diferentes técnicas de optimización o de aproximación para solucionar el problema.

REFERENCIAS

[1] JARAMILLO G, Andrés; RESTREPO C, Jorge. Aplicación de la programación dinámica para resolver el problema simple de balanceo de línea de ensamble. Scientia et Technica Año XV,

No 43, Octubre de 2009. Universidad Tecnológica de Pereira. ISSN 0122-1701
[2] BAZARAA, JARVIS. Programación lineal y flujo de redes. Editorial Limusa– Noriega editores. 1994

[3] CAPACHO B, L, PASTOR, R. Generación de secuencias de montaje y equilibrado de línea, Universidad Politécnica de Catalunya, Abril 2004

[4] EGUIA S, Ignacio. Método de asignación y secuenciación de tareas en el diseño de una cadena de montaje monomodelo usando programación dinámica. Universidad de Sevilla

[5] HELD M, KARP R.M, SHARESHIAN R, Assembly line balancing — dynamic programming with precedence constraints, Operation Research. Vol 11 442–459. 1963

[6] HILLIER, LIEBERMAN. Investigación de operaciones. Mc Graw Hill. 2001

[7] JARAMILLO G, Andrés, Aplicación de la programación dinámica para resolver el problema de balanceo de línea de ensamble simple, Tesis de grado Ingeniero Industrial, Universidad Tecnológica de Pereira, Marzo de 2009.

[8] KAO, E.P.C, QUEYRANNE, On dynamic programming methods for assembly line balancing, Operations Research. Vol 30 (2) 375–390. 1982

[9] PRAWDA, W Juan. Métodos y modelos de investigación de operaciones. Editorial Limusa. 1976

[10] SCHOLL, A. Balancing and sequencing of Assembly lines. Physica-Verlag, 1999.