APLICACIÓN DE ALGUNOS MÉTODOS EXÀCTOS Y HEURÌSTICOS PARA RESOLVER EL PROBLEMA DE BALANCEO DE LINEA SIMPLE

Autores

JAVIER ANDRES PÓVEDA ZAPATA

10.004.424

CARLOS HERNANDO FLOREZ HURTADO

9.861.610

UNIVERSIDAD TECNOLÓGICA DE PEREIRA

FACULTAD DE INGENIERÍA INDUSTRIAL

PEREIRA

2009

APLICACIÓN DE ALGUNOS MÉTODOS EXACTOS Y HEURISTICOS PARA RESOLVER EL PROBLEMA DE BALANCEO DE LINEA SIMPLE

Autores

JAVIER ANDRES PÓVEDA ZAPATA

10.004.424

CARLOS HERNANDO FLOREZ HURTADO

9.861.610

Trabajo de Grado para optar el título de ingeniero industrial

Asesor:

JORGE HERNAN RESTREPO CORREA

Mcs Investigación De Operaciones y Producción

UNIVERSIDAD TECNOLÓGICA DE PEREIRA

FACULTAD DE INGENIERÍA INDUSTRIAL

PEREIRA

2009

Nota de aceptación

Firma del presidente

Firma del jurado

Firma del jurado

Pereira 26 de Agosto del 2009

DEDICATORIA DE JAVIER ANDRES POVEDA ZAPATA:

A mi familia por su apoyo incondicional y vehemente en todos estos años de arduo sacrificio, a Dios por guiar mi vida en los momentos de infortunio y desesperanza en los que me colmo de sabiduría, a mis profesores por compartir sus conocimientos durante mi formación como profesional, y a mis amigos mas cercanos por sus consejos y aportes valiosos.

DEDICATORIA DE CARLOS HERNANDO FLOREZ:

A mi familia la cual me apoyo con esfuerzo durante todos estos años de estudio, a los profesores por sus enseñanzas y paciencia, a mi pareja Lorena García y a mi hija María Fernanda que son la luz que guían y motivan día a día mi vida y me permiten seguir luchando, a Dios por la fortaleza y sabiduría que me brindo, por último a todos mis amigos que de una u otra forma me colaboraron cada semestre y en especial con este proyecto.

RESUMEN

Este documento investigativo pretende dar solución a un ejemplo particular de balanceo de línea simple, el cual se estudiará a través de dos tipos de problemas denominados SALBP-1 y SALBP-2, por medio de un método heurístico conocido como COMSOAL y un método exacto como lo es el BRANCH & BOUND.

En el documento se hace un breve resumen sobre los métodos más usados para resolver el problema de balanceo, y se describe con claridad los tipos de líneas y sus características.

El ejemplo es resuelto por los dos métodos citados; en COMSOAL los pasos para la resolución del caso SALBP-1 y SALBP-2 son descritos en forma detallada, mientras que el BRANCH AND BOUND por ser más extenso se apoya del programa WINQSB versión 1.0. Finalmente sus metodologías son comparadas entre si y de esta forma se concluye cual método es el más práctico.

ABSTRACT

This document pretend to give solution to an particular example of simple line balancing, which it's will study through two types of problems name SALBP-1 and SALBP-2, through, a heuristic method known as COMSOAL and exact method known as BRANCH AND BOUND.

In the document, it is make a brief abstract about the methods most used to solve the balancing problem and it's describe with clearness the types of lines and theirs characteristics.

Example is solved by two methods named; in COMSOAL the steps for solving of the case SALBP-1 and SALBP-2 are described in form detailed, while method BRANCH AND BOUND by to be more extended, it's support of the program WINQSB version 1.0. Finally their methodologies are compared and so to conclude what method is the most practical.

TABLA DE CONTENIDO

1. LISTA DE FIGURAS

2. LISTA DE TABLAS

3. GLOSARIO

ALGORITMO: es una lista bien definida, ordenada y finita de operaciones que permite hallar la solución a un problema. También se define como un conjunto de operaciones y procedimientos que deben seguirse para resolver un problema.

BANDA TRANSPORTADORA: consiste en 2 o más poleas con material circulando entre ellas, una o ambas poleas están motorizadas (polea motriz) y hacen mover el material de un lugar a otro, también existen las bandas manuales donde el producto es puesto sobre la banda y una persona hace mover el material (polea de arrastre).
En las industrias existen generalmente dos tipos de bandas:

- Las que se utilizan para transportar productos en general.
- Las bandas que se utilizan para transportar productos a granel.

CUELLO DE BOTELLA: son actividades que detienen o disminuyen el flujo normal de un proceso productivo, haciendo aumentar los tiempos de espera y disminuyendo la eficiencia.
Los cuellos de botella se pueden presentar en las personas o en las máquinas, debido a la poca capacitación y entrenamiento en el caso de las personas y en las máquinas debido a la falta de mantenimiento.

FUNCION OBJETIVO: es una expresión matemática que representa el objetivo del problema, en el caso bidimensional consiste en maximizarla o minimizarla.

JUSTO A TIEMPO (Just in time): es una filosofía industrial, utilizada por el sistema productivo de Toyota y consiste en la reducción o eliminación de desperdicios en compras, fabricación, distribución y funciones de apoyo (trabajo de oficina), considerando desperdicio a todo aquello adicional a los recursos mínimos necesarios de máquinas, materiales y mano de obra que agregan valor al producto.

PATINADOR: es la persona encargada de transportar materiales, insumos, producto en proceso y otros requerimientos de una estación a otra.

VARIABLE BOOLEANA: en computación es aquella que puede representar valores binarios, es decir, toma valores de 0 o 1.

4. INTRODUCCION

El problema del balanceo de línea puede ser resuelto por medio de la aplicación de diferentes algoritmos ya sean heurísticos, exactos o dinámicos, el siguiente documento solo se preocupa por el método exacto Branch and Bound y el método heurísticos Comsoal, con el fin de conseguir buenos resultados en las líneas de producción. Como es sabido una línea bien equilibrada trae no solo reducción en los costos de operación, sino también un mejor nivel de calidad y por lo tanto un aumento en la satisfacción de los clientes, por otro lado la competitividad se mejora notablemente, viéndose reflejado este incremento en las ventas y en la imagen de las empresas.

Se debe reconocer los dos tipos de problema de balanceo de línea, los cuales son: el simple ó SALBP (simple assembly line balancing problem) y el general ó GALBP (general assembly line balancing problem). Los métodos del presente documento se aplican para la solución de problema simple de balanceo, el cual es el objeto de estudio en particular.

En este tipo de problema se consideran líneas de producción simples en las que las estaciones son colocadas en serie y solo existe un solo tipo de producto con tiempos conocidos, además las estaciones pueden realizar cualquier tipo de tarea y las tareas pueden ser realizadas en cualquier estación, los tiempos en el problema simple son deterministicos y desde luego conocidos con anticipación.

5. ANTECEDENTES DE LA INVESTIGACIÓN

5.1 Planteamiento del problema

5.1.1 Diagnóstico del problema

En un mundo cada vez más competitivo, en donde el mercado actual es cada vez más exigente, las empresas modernas se ven obligadas a evolucionar en todos sus aspectos funcionales. En la actualidad, por efecto de la globalización, las industrias deben mejorar la competitividad, y ésta conlleva el mejoramiento de los procesos de manufactura, es decir la optimización de las variables y recursos de la empresa, para reducir costos, mejorar calidad y eficiencia; por ello adquiere importancia el estudio del balanceo de línea de ensamble.

Se puede observar que la eficiencia productiva juega un papel muy importante en el crecimiento de las empresas, es por tal razón que se debe procurar por el buen uso de los recursos en el proceso de producción, de tal manera que las empresas deben preocuparse por conocer, estudiar y mejorar el comportamiento de las diferentes variables involucradas al interior de su sistema de manufactura, específicamente, la línea de ensamble, ya que es aquí en donde se presentan diferentes problemas: minimizar estaciones, minimizar tiempo ocioso, maximizar flujo de producción, minimizar tiempos de ciclo; en otras palabras asignar de manera óptima los recursos con que se cuenta, lo cual genera en la empresa una disminución en los costos de operación y en consecuencia, aumento en la competitividad. Los problemas de este tipo se conocen como balanceo de línea.

El estudio de los sistemas de producción y operaciones, es muy importante, dentro de la ingeniería industrial, que requiere en la actualidad la solución de problemas que presentan las empresas en el área de producción, utilizando diversas herramientas y métodos para solucionarlos. Esta investigación se centrará en el balanceo de una línea de ensamble simple, la cual es la línea más común en la industria moderna.

5.1.2 Formulación del problema

¿Qué métodos exactos y heurísticos se pueden aplicar para dar solución al problema de balanceo de línea simple (SALBP)?

5.1.3 Sistematización del problema

¿Cómo se puede resolver el problema de balanceo de línea simple SALBP-1 y SALBP-2 por medio del COMSOAL?

¿Cómo se puede resolver el problema de balanceo de línea simple SALBP-1 y SALBP-2 por medio del BRANCH AND BOUND utilizando el programa Winqsb?

¿Qué método brinda una mejor eficiencia para el problema de minimizar el número de estaciones dado un tiempo de ciclo?

¿Qué método brinda una mejor eficiencia para el problema de reducir el tiempo de ciclo dado un número de estaciones?

6. JUSTIFICACION

Conscientes de la necesidad de solucionar el problema de balanceo de líneas que se presenta en la industria mundial, se pretende investigar diferentes metodologías adecuadas para solucionar dicho problema, aportando así el conocimiento recopilado en la investigación de este documento como texto de consulta para las empresas que necesitan implementar una solución adecuada en sus problemas de balanceo.

Existen muchas clases de problemas de balanceo de líneas de ensamble, en esta investigación se tratará el problema del balanceo de línea simple de ensamble, que es el más común en la actualidad. Es importante realizar esta investigación pues es necesario confrontar los conocimientos teóricos adquiridos a lo largo de la carrera y tratar de integrarlos por medio de la aplicación conceptual en este estudio. Aquí se puede observar la integración de conocimientos de diferente tipo, tanto en ciencias básicas, como en temas del área de producción e investigación de operaciones. Esto es importante porque permite tener una visión amplia y global de la situación y resolver y dar soluciones a los problemas de manera sistémica y no de forma aislada.

Es necesario que la universidad, específicamente, la facultad de ingeniería industrial tenga como finalidad trabajar de la mano con el entorno, trabajar con las empresas, dirigir procesos de cambio, innovación y mejoramiento de los procesos en las empresas, que se enfoque en resolver los problemas que las organizaciones necesitan que se resuelvan, y que se genere un conocimiento dinámico, es decir, que a medida que cambie el entorno, sus problemas y necesidades, la facultad se interese en resolver dichos problemas, siendo flexible, enseñando teorías y aplicaciones de uso actual.

7. DISEÑO METODOLÓGICO

Se define con claridad el problema de balanceo de línea simple, se clasifica de acuerdo a lo que se busca optimizar y se plantea el modelo matemático para un ejemplo específico, brindando una solución por medio de los modelos Comsoal y Branch and Bound (winqsb) investigados en este texto, por último se hará una comparación y análisis de los métodos aplicados, apoyados en una revisión bibliográfica referente al tema.

8. OBJETIVOS DE LA INVESTIGACION

8.1 OBJETIVO GENERAL

Solucionar el problema de balanceo de línea simple tipo SALBP-1 y SALBP-2 con el método COMSOAL y BRANCH & BOUND (Winqsb Versión 1.00)

8.2 OBJETIVOS ESPECÍFICOS

- Definir el problema de balanceo de línea.
- Clasificar el problema.
- Plantear el modelo matemático del problema.
- Solucionar con COMSOAL.
- Solucionar con BRANCH AND BOUND (Winqsb Version 1.00).
- Comparar y analizar los resultados.

9. MARCO CONCEPTUAL

9.1 DEFINICIONES RELATIVAS A LAS LÍNEAS DE ENSAMBLE

CARGA DE TRABAJO: es el conjunto Sk de tareas asignadas a la estación k.

DIAGRAMAS DE PRECEDENCIA: se usan para representar las relaciones de precedencia.

EFICIENCIA: Es la relación existente entre el vector insumos (cantidad, calidad, espacio y tiempo) y el vector productos, durante el subproceso estructurado, de conversión de insumos en productos.

EQUILIBRADO DE LÍNEAS: el equilibrado de líneas busca asignar mejor los recursos de que se dispone en un proceso de manufactura a través de una buena distribución de las tareas en un determinado numero de estaciones, se pueden considerar diferentes circunstancias a la hora de solucionar un problema de este tipo como por ejemplo un numero mínimo de estaciones o un mínimo tiempo de ciclo en el que se puede realizar una tarea o varias de ellas por supuesto teniendo en cuenta las relaciones de precedencia entre estas, si es que existen.

ESTACIÓN: es la parte k de la línea de montaje en donde se ejecutan las tareas; pueden estar compuestas por un operador (humano o robotizado), cierto tipo de maquinaria y equipos o mecanismos de proceso especializados.

RELACIONES DE PRECEDENCIA: están definidas por las restricciones sobre el orden en el cual las operaciones pueden ser ejecutadas en la línea de montaje. De esta forma, una tarea no puede procesarse hasta que no se hayan procesado todas las que le preceden de forma inmediata.

RESTRICCIONES: es cualquier elemento que evita que una organización genere ganancias, dentro de estas se encuentran dos tipos:
Las restricciones físicas: son aquellas que tienen que ver con el mercado, las personas, materiales, piezas o maquinas.
Las restricciones de política: son aquellas que tratan de reglas, procedimientos y sistemas de evaluación.

TAREA: es una unidad de trabajo indivisible j que tiene asociado un tiempo de proceso t_j. El trabajo total requerido para manufacturar un producto en una línea se divide en un conjunto de *n t*areas $V = \{1,..., j,..., n\}$.

TIEMPO DE CADA ESTACIÓN: es la suma de los tiempos de todas las tareas asignadas a una estación:

$$t(S_k) = \sum_{j \in S_k} t_j \ .$$

TIEMPO DE CICLO (C): es el tiempo disponible en cada estación para completar las tareas asignadas para una unidad de producto. Puede ser el tiempo máximo o el tiempo promedio disponible para cada ciclo de trabajo.

TIEMPO DISPONIBLE EN LA ESTACIÓN: es el tiempo que queda en la estación, después de asignar una o varias tareas, sin exceder el tiempo de ciclo.

TIEMPO OCIOSO: es la diferencia entre el tiempo de ciclo y el tiempo de estación:

$$c - t(S_k).$$

TIEMPO MUERTO TOTAL O DEMORA DEL BALANCE: es la cantidad total de tiempo ocioso en la línea, debido a una asignación desigual de las tareas en cada estación.

$$D = m \cdot C - \sum_{i=1}^{n} t_i$$

9.2 DEFINICIONES RELATIVAS AL COMSOAL

COMSOAL: por sus siglas en ingles Computer Method of sequencing operations for an assembly line. Es un método computarizado que permite asignar tareas a las estaciones de trabajo, realizando cientos de iteraciones y entregando la mejor solución en poco tiempo.

HEURÍSTICA[1]: la palabra heurística como tal se refiere específicamente al poder que tiene un sistema de adaptarse a las condiciones del entorno e innovar de manera eficiente para sus propósitos finales.

La heurística se basa fundamentalmente en métodos de carácter exploratorio para resolver un problema en particular en los cuales las soluciones se obtienen gracias al trabajo logrado para obtener un resultado final, no obstante la solución no es necesariamente la optima como en los métodos exactos, sino que puede ser una solución aproximada.

[1] http://es.wikipedia.org/wiki/Heur%C3%ADstica,noviembre 2008

LISTA A: lista de tareas con sus tiempos de operación, tareas predecesoras y número de predecesoras.

LISTA B: conjunto de tareas de la lista A, que tienen cero predecesoras.

LISTA C: son las tareas de la lista B, que cumplen con el tiempo de ciclo, si esta completo o con el tiempo disponible en la estación.

9.3 DEFINICIONES RELATIVAS AL BRANCHAND BOUND

BRANCH AND BOUND: Técnica conocida por sus aplicaciones a los problemas de programación entera, la técnica se fundamenta en la idea de dividir y vencerás.

RAMIFICACION: consiste en dividir el problema inicial en dos o más problemas para hacer más fácil su tratamiento.

SONDEO O ACOTAMIENTO: es el acotamiento de la mejor solución en el subconjunto y después eliminando los subconjuntos cuya cota muestre que no es la indicada.

RELAJACION CONTINUA: consiste en encontrar una solución inicial que arroja valores continuos, para así tener un punto de partida y poder comenzar a tratar el problema, esta solución se convierte en una cota superior en el caso que se este maximizando.

INCUMBENTE: solución de apoyo o temporal encontrada en el desarrollo de un problema.

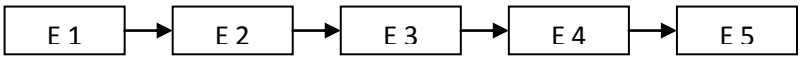
10. MARCO TEORICO

10.1 CARACTERIZACION DE LAS LINEAS DE ENSAMBLE

De acuerdo a la arquitectura de la línea

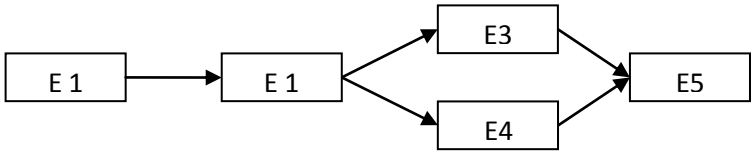
Línea serial: esta conformada por estaciones simples donde el producto pasa de una estación realizándole los respectivos cambios, se puede usar una banda transportadora para el movimiento del producto.

Figura 1. Línea serial



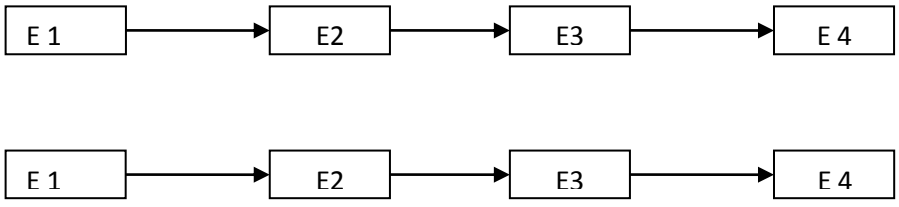
Línea con estaciones en paralelo: es una línea serial en la cual algunas estaciones tienen estaciones en paralelo con el fin de disminuir su carga de trabajo, evitar los cuellos de botella, y reducir el tiempo de una tarea cuando esta es mayor que el tiempo de ciclo. Las estaciones en paralelo poseen los mismos equipos y operarios que la estación original de la línea**.**

Figura 2. Línea con estaciones en paralelo



Líneas paralelas: son varias líneas colocadas en paralelo con el fin de procesar diferentes productos o familias de productos por cada línea, aquí se encuentra un nuevo problema y es decidir cuantas líneas y como distribuir los equipos y fuerza de trabajo.

Figura 3. Líneas paralelas

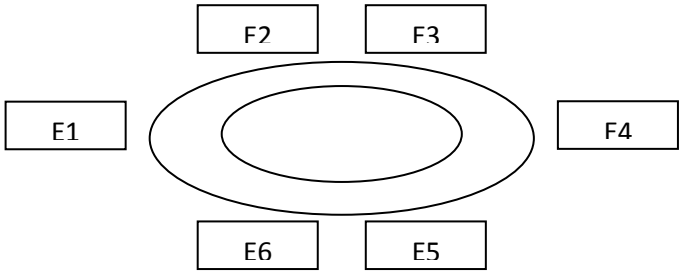


Línea de dos lados: son líneas seriales paralelas, las cuales operan al tiempo un mismo producto, las líneas están en capacidad de realizar la misma tarea u otra. Un ejemplo típico de estas líneas es la de automóviles donde se deben realizar las mismas taras en ambos lados (puertas, espejos, vidrios, etc).

Figura 4. Línea de dos lados[2]

Líneas circulares /cerradas: en estas líneas las piezas van circulando, mientras los operarios o robots las van tomando y las procesan y vuelven a liberarlas en la línea.

Figura 5. Líneas circulares cerradas.

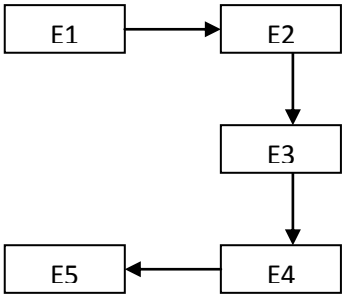


Línea en forma de U: este tipo de líneas se crearon pensando en el justo a tiempo, aquí hay más flexibilidad debido a que los operarios pueden moverse

[2] Foto tomada de www.velocidadmaxima.com/forum/showthread.php?t=1187,marzo 2009

de una estación a otra, además esta configuración puede resultar en un mejor balanceo en la carga de las estaciones, dado que el número de combinaciones tareas-estación es más grande.

Figura 6. Línea en forma de U



De acuerdo al tipo de flujos de las piezas

Sincrónicas: todas las estaciones tienen un tiempo de ciclo común, por tal motivo los productos pasan de una estación a otra al mismo tiempo evitando los buffer de entrada.

Asincrónicas: aquí no se tiene un tiempo de ciclo común, por este motivo existe buffer entre las estaciones y se crea el problema de en cual estación colocar los almacenamientos de productos en proceso.

Líneas de alimentación: este tipo posee una línea principal donde se ensamblara el producto final, pero también posee otras líneas secundarias que harán subensambles que alimentaran con productos intermediarios a la principal, aquí se tendrá en cuenta tanto balancear la línea principal como sincronizar las líneas secundarias.

De acuerdo al tipo de operador

Líneas manuales: pueden ser automatizados o no, y son operadas por humanos.

Líneas robotizadas: son totalmente automatizadas y los operadores son robots, en este tipo de líneas hay que planificar el procesamiento de las tareas en las estaciones y planificar las actividades de los robots.

De acuerdo a la disciplina de entrada de las piezas a la línea

Línea de entrada fija: las piezas entran a intervalos iguales de tiempo y si la línea es sincrónica corresponderá al tiempo de ciclo.

Línea de entrada variable: la tasa de entrada de las piezas a la línea es variable.

Línea de ensamble o montaje: las líneas de ensamble están constituidas por un número determinado de estaciones en las cuales se realizan una serie o conjunto de tareas finitas que pueden estar o no precedidas entre si las cuales son requeridas para la elaboración de un producto específico. Este tipo de configuración fue introducido por Eli Whitney en la era industrial, a quien se le atribuye la invención del sistema de manufactura americano en 1799 basado en las ideas de la división del trabajo y la tolerancia en ingeniería.

10.2 TIPOS DE PROBLEMAS DE BALANCEO DE LINEA

Teniendo en cuenta los tipos de líneas, existirá un problema para resolver en cada uno, así aparece dos tipos de problemas de balanceo de línea, enunciados por Baybars[3]:

Los SALBPS (Simple assembly line balancing problem) o problema simple de balanceo de línea.
Este es el tipo de problema de balanceo mas simple y sencillo, aquí se tiene en cuenta que todas las maquinas son idénticas y pueden procesar cualquier tarea, pero las tareas no se pueden procesar en paralelo pues existen relaciones de precedencia que hacen que para que una tarea sea procesada es necesario que se procese primero otra, además los tiempos de procesamiento de las tareas son conocidos para cada maquina.

10.2.1 Clasificación de los SALBPS

Los problemas SALBPS se pueden clasificar de diferentes maneras dependiendo de su objetivo, así:

SALBP -1: El problema reside en hallar una solución factible que permita minimizar el número de estaciones, dado un tiempo de ciclo. Este tipo se da cuando se va a hacer un nuevo montaje y la demanda puede ser estimada.

SALBP-2: Busca minimizar el tiempo de ciclo dado un número de estaciones fijas, aquí se supone que la línea existe.

SALBP-E: este problema busca maximizar la eficiencia de la línea, es decir busca minimizar el producto de m (numero de estaciones) por c (tiempo de ciclo).

---

[3] CAPACHO L, MORENO R, Generación de secuencias de montaje y equilibrado de líneas, Universidad Politécnica de Catalunya, Abril 2004.

SALBP –F: se busca una solución factible dado el tiempo de ciclo y el numero de estaciones, es decir, se debe mirar si una línea funciona con m estaciones y un tiempo de ciclo c.

10.2.2 Modelación matemática de los SALBPS[4]

Modelación matemática SALBP-1

$x_{ij} = 1$ *si operación i se hace en estación j*
$y_j = 1$ *si existe estación j*

$$Min\ Z = \sum_{j=1}^{M_{max}} y_j$$

$$s.a$$

$$(1)\quad \sum_{i=1}^{N} t_i \cdot x_{ij} \le C \cdot y_j \qquad j = 1,\ldots, M_{\max}$$

$$(2)\quad \sum_{j=1}^{M_{max}} x_{ij} = 1 \qquad i = 1,\ldots, N$$

$$(3)\quad \sum_{j=1}^{M_{max}} j \cdot x_{kj} \le \sum_{j=1}^{M_{max}} j \cdot x_{ij} \qquad \forall k \prec i$$

$$(4)\quad y_{j+1} \le y_j \qquad j = 1,\ldots, M_{\max} - 1$$

$$x_{ij} = \{0,1\} \quad \forall (i,j); \qquad y_j = \{0,1\} \quad \forall (j)$$

Modelación matemática SALBP-2

$x_{ij} = 1$ *si operación i se hace en estación j*
C *tiempo de ciclo*

$$Min\ C$$

$$s.a$$

$$(1)\sum_{i=1}^{N} t_i \cdot x_{ij} \le C \qquad j = 1,\ldots, M$$

$$(2)\sum_{j=1}^{M} x_{ij} = 1 \qquad i = 1,\ldots, N$$

$$(3)\sum_{j=1}^{M} j \cdot x_{kj} \le \sum_{j=1}^{M} j \cdot x_{ij} \qquad \forall k \prec i$$

$$x_{ij} = \{0,1\} \quad \forall (i,j); \qquad C \in \Re^{+}$$

[4] http://racero.us.es/Asignaturas/Secuenciacion/EQUILIBRADO.PPT (septiembre2008)

Los GALBPS (General assembly line balancing problem) o problema general de equilibrados de línea.

Los GALBPS son aquellos problemas que no están dentro de la categoría de los SALBP, este tipo de problemas se diferencian de los SALBP porque son mas reales.
Aquí se tiene en cuenta las estaciones en paralelo, los modelos mixtos, los tiempos de proceso variable, entre otros.

10.2.3 Clasificación de los GALBPS

Dentro de los GALBPS se identifican cuatro tipos que son los más utilizados:

UALBP (Problemas de equilibrado de línea tipo U)
Este tipo de problema se destaca por utilizar líneas de ensamble tipo U, donde se pueden trabajar dos productos en diferentes puntos de la línea. Estos a su vez se clasifican dependiendo de algunas variables de decisión como:

UALBP 1 (Minimiza el numero de estaciones)

UALBP 2 (Minimiza el tiempo de ciclo)

UALBP E (Maximiza la eficiencia de la línea)

MALBP (Problema de equilibrado de línea de modelos mixtos)
En estos problemas se considera diferentes modelos de un mismo producto, teniendo en cuenta conjuntos de tareas comunes para todos los modelos. Los MALBP también se clasifican en MALBP 1, MALBP 2 y MALBP E

RALBP (Problema de equilibrado de líneas robotizadas)
En este tipo de problema existe una combinación entre la asignación de tareas y la asignación de un grupo de robots a las estaciones de trabajo, con el fin de optimizar los recursos de la línea.

MOALBP (Problema de equilibrado de líneas con objetivos múltiples)
Los MOALBP tienen en cuenta objetivos múltiples como por ejemplo minimizar el número de estaciones y el coste. Rekiek (2002, Pg 163-174) dice que la mayoría de problemas de equilibrado de línea presentan objetivos múltiples.

10.3 METODOS DE RESOLUCION DE EQUILIBRADOS DE LINEAS

En la literatura encontramos dos métodos de resolución de problemas de equilibrados los cuales son:

- Los métodos exactos.
- Los métodos heurísticos.

10.3.1 Métodos exactos

Los métodos exactos garantizan una solución óptima usando programación matemática y algoritmos exactos para la exploración de grafos[5]. Algunos de estos métodos son:

Programación exacta

Programación lineal[6]

La programación lineal es apenas una pequeña fracción de la teoría matemática, más conocida como optimización, por medio de esta técnica se busca obtener el mayor beneficio en sistemas de diferentes tipos: Económicos así como sociales entre otros.

Para que exista un problema de programación lineal, todas las funciones, objetivo y restricciones deben ser lineales.

Un problema de Programación Lineal consiste en optimizar (maximizar o minimizar) la función:

$$z = F(x_1, x_2, \ldots, x_n) = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

Sujeto a:

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq = \geq b$$

$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq = \geq b_2$$

. . .

$$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leq = \geq b_m$$

$$x_1, x_2, \ldots, x_n \geq 0$$

A la función $z = F(x_1, x_2, \ldots, x_n) = c_1x_1 + c_2x_2 + \ldots + c_nx_n$ se le denomina función objetivo o función criterio.

Los coeficientes $c_1, c_2, \ldots, c_n$ son números reales y se llaman coeficientes de beneficio o coeficientes de costo. Son datos de entrada del problema.

---

[5] https://upcommons.upc.edu/pfc/bitstream/2099.1/2587/1/31026-1.pdf,septiembre 2008

[6] http://descartes.cnice.mec.es/materiales_didacticos/prog_lineal_lbc/definicion_pl.htm,septiembre 2008

$x_1, x_2, \ldots, x_n$ son las variables de decisión (o niveles de actividad) que deben determinarse.

Las desigualdades $a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \leq b_i$, con $i = 1, \ldots, m$ se llaman restricciones.

Los coeficientes a_{ij}, con $i = 1, \ldots, m$ y $j = 1, \ldots, n$ son también números reales conocidos y se les denomina coeficientes tecnológicos.

El vector del lado derecho, es decir los términos b_i, con $i = 1, \ldots, m$, se llama vector de disponibilidades o requerimientos y son también datos conocidos del problema.

Las restricciones $x_j \geq 0$ con $j = 1, \ldots, n$ se llaman restricciones de no negatividad.

Al conjunto de valores de $(x_1, x_2, \ldots, x_n)$ que satisfacen simultáneamente todas las restricciones se le denomina región factible. Cualquier punto dentro de la región factible representa un posible programa de acción.

La solución óptima es el punto de la región factible que hace máxima o mínima la función objetivo.

Programación lineal Binaria[7]

Son modelos de programación matemática que se pueden representar de forma lineal pero cuyas variables pueden tomar únicamente los valores de 0 o 1. White utilizo la programación lineal binaria para dar solución al problema SALBP y su forma de modelarlos fue la siguiente

Función Objetivo: MIN Z=

$$[\text{MIN}]z= \sum_{i \in F} \sum_{j=m\min+1}^{m\max} c_j * x_{i,j}$$

[7] http://www.investigacion-operaciones.com/Libro/Programacion%20Entera.pdf

Restricciones:

$$\sum_{j=1}^{m} x_{i,j} = 1$$

i=1……n

La anterior restricción nos indica o nos muestra que una tarea solo puede ser asignada a una estación de trabajo: criterio de tareas indivisibles.

$$\sum_{i=1}^{n} t_i * x_{i,j} \leq T\max$$

j =1……m

La anterior restricción limita el tiempo de ciclo en cada estación de trabajo.

$$x_{i,k} \leq \sum_{j=1}^{k} x_{h,j}$$

K= 1…..m i=1….n $h \in p$

La restricción descrita muestra que la tarea i es precedente de la tarea h

Con:

$$c_{j+1} \geq M * c_j$$

j= (mmin+1)…… (mmax-1)

$$x_{i,j} \in \{0,1\} \; \forall \; i,j$$

Donde:

$X_{i,j}$: Es la variable que indica si la tarea i se realiza en la estación j, si esto sucede toma el valor de 1, de lo contrario toma el valor de 0.

tj: tiempo en que se realiza la tarea i.

Tmax: Valor máximo del tiempo de ciclo.

M: Valor suficientemente grande.

F: conjunto de tareas sin sucesoras.

p (i): conjunto de tareas predecesoras inmediatas de la tarea i.

n: numero de tareas que forman la línea de montaje.

m: numero de estaciones que componen la línea.

mmin: número mínimo de estaciones.

mmax: numero máximo de estaciones.

cj: son coeficientes de la función objetivo que fuerzan al modelo a asignar las tareas como primera medida a las primeras estaciones, antes que asignarla a las ultimas estaciones. Por tal motivo al ser más grande este coeficiente entre mas alejado se encuentre la estación hace que se penalice estas estaciones, evitando que se les asigne tareas mientras exista otras estaciones anteriores disponibles[8].

Método de Thangavelu

Este algoritmo es parecido al de programación lineal binaria de White, pero cambiando su función objetivo cj para evitar inestabilidades numéricas presentadas en algunos casos, especialmente en aquellos que tenían gran número de tareas.

Valero

Este autor manejo programación lineal entera pero que tenía en cuenta las incompatibilidades entre tareas, considera diferentes tipos de estaciones y los recursos que consumen.

Programación dinámica

La programación dinámica permite solucionar problemas en los que se toman decisiones en etapas sucesivas, las decisiones que se toman en una etapa afectan el desarrollo futuro del sistema, tanto a los estados como a las decisiones que se tomen mas adelante.

[8] https://upcommons.upc.edu/pfc/bitstream/2099.1/2587/1/31026-1.pdf,agosto 2008

La programación dinámica no sigue un modelo estándar como si ocurre con la programación lineal, por tal motivo para cada problema es necesario definir los componentes que lo caracterizan.

La forma de solucionar un problema de programación dinámica ocurre desde la última etapa y en orden sucesivo hasta la primera, cuando se acaba de analizar la primera etapa es donde se obtiene el óptimo del problema[9].

La programación dinámica es de gran utilidad en problemas de optimización, en estos tipos de problema se pueden dar diferentes soluciones y lo que se busca es encontrar aquella que de un valor optimo, que puede ser un valor máximo o un valor mínimo. Este tipo de soluciones se basan en el principio de Bellman (1957) y que dice: "En una secuencia de decisiones óptima toda subsecuencia ha de ser también optima"

En grandes líneas, el diseño de un algoritmo de Programación Dinámica consta de los siguientes pasos:

1. Planteamiento de la solución como una sucesión de decisiones y verificación deque ésta cumple el principio de óptimo.

2. Definición recursiva de la solución.

3. Cálculo del valor de la solución óptima mediante una tabla en donde se almacenan soluciones a problemas parciales para reutilizar los cálculos.

4. Construcción de la solución óptima haciendo uso de la información contenida en la tabla anterior.

Un algoritmo reconocido dentro de la programación dinámica es el modelo de Held (1963) que minimiza el numero de estaciones para un tiempo de ciclo dado, aquí se tiene un subconjunto factible conformado por n tareas que pueden ser ejecutadas en algún orden sin que las otras tareas hayan sido ejecutadas[10].

10.3.2 Métodos heurísticos

[9] http://www.eumed.net/libros/2006c/216/1j.htm,agosto 2008

[10] https://upcommons.upc.edu/pfc/bitstream/2099.1/2587/1/31026-1.pdf, agosto 2008

Los métodos Heurísticos entregan una solución alejada del óptimo, pero en ocasiones se puede llegar al óptimo.

Dentro de estos métodos tenemos:

- Heurísticas de una sola pasada.
- Reglas de Backtracking
- Aproximación a partir de algoritmos exactos.
- Heurísticas de composición.

Aquí daremos una breve introducción de estas metodologías y los algoritmos que los conforman.

Heurísticas de una sola pasada

Se basan en reglas de decisión simples. Dentro de este método encontramos los siguientes algoritmos:

- El algoritmo de Tongue.
- El algoritmo de Moodie Young.
- El algoritmo de Helgeson and Birnie[11].

Se explicara cómo funciona este último algoritmo

Paso 1: Se debe calcular el peso posicional de cada elemento, que consiste en sumar el tiempo de todas las tareas que le siguen, incluyéndose la tarea en cuestión.

Paso 2: Se debe calcular el tiempo de ciclo, que se puede dar por dos motivos:

- Cumplir con una demanda esperada.
- Minimizar el tiempo ocioso en la línea de ensamble.

Paso 3: Se asigna las tareas a las estaciones de trabajo comenzando por el que tiene mayor peso posicional, teniendo en cuenta las precedencias y que exista tiempo de ciclo disponible, si la tarea a asignar no cumple con el tiempo de ciclo se pasa a otra de menor peso que si lo cumpla.

[11] http://www.utp.edu.co/php/revistas/ScientiaEtTechnica/docsFTP/18521325-330.pdf

Reglas de Backtracking

Backtracking (retroceso): En forma simple es una técnica de programación, la cual tiene diferentes caminos para escoger, cuando se llega al final del proceso y la solución no cumple con las condiciones dadas, se vuelve atrás para buscar un nuevo camino que permita buscar una nueva solución que puede ser la optima.
El Backtracking permite hacer una búsqueda sistemática a través de un espacio de posibles soluciones, por tal motivo este método crea una solución candidata s y hace dos cosas:

1. Revisa si s es una solución, si lo es, la utiliza dependiendo del problema que se esta tratando.

2. Si no lo es, crea todas las posibles extensiones y aplica el algoritmo en cada una de ellas.

El algoritmo genérico del Backtracking es el siguiente:

$B_T(A,K)$

- If SOLUCION?(A,K)
- Then PROCESAR SOLUCION (A,K)
- Else for each c Є SUCESORES (A,K)
- Do A[K]= c
- B_T=(A,K+1)
- If TERMINAR?
- Then RETURN

Donde:

SOLUCION? ( ): Es una función que retorna verdadero si su argumento es una solución.

PROCESAR SOLUCION ( ): Depende del problema que se quiere resolver.

SUCESORES ( ): Es una función que dado un candidato genera todos los candidatos que son extensiones de este.

TERMINAR? : Es una variable global Booleana inicialmente falsa, pero que se puede volver verdadera por PROCESAR SOLUCION ( ), en caso de que solo se quiera encontrar una solución.

Dentro de las reglas de Backtracking también encontramos las heurísticas de Hoffman (1963 Pág. 551- 562) y el algoritmo de MALB de Dar El (1973, Pág. 343-356).

Para Bartak el backtracking simple realiza algún tipo de técnica de consistencia, pudiendo ser visto como una combinación de generate and test y una fracción de arco-consistencia. El algoritmo BT prueba arco-consistencia entre las variables asignadas, verifica la validez de las restricciones considerando las asignaciones parciales; como a las variables se asigna un único valor, es posible verificar únicamente los arcos que contienen la última variable fijada, si su dominio se reduce, entonces la restricción correspondiente no es consistente, y el algoritmo retrocede a una nueva asignación.

Aproximación a partir de algoritmos exactos

Estos algoritmos parten de procedimientos exactos a los cuales se les ha restringido el tiempo de búsqueda de la solución óptima. Dentro de este grupo se encuentra el Algoritmo de HELD, TALBOT AND PATTERSON y el FABLE de Jonson[12].

A diferencia de las heurísticas, que usualmente sólo encuentran soluciones razonablemente buenas en tiempos razonablemente rápidos, lo que se busca aquí es encontrar soluciones que está demostrado son de calidad y cuyos tiempos de ejecución están acotados por cotas conocidas. Los algoritmos de aproximación están siendo cada vez más utilizados para resolver problemas donde los algoritmos exactos de tiempo polinomial son conocidos pero demasiado costosos debido al tamaño de la entrada.

Heurísticas de composición

Consiste en una composición de reglas de decisión, dentro de la cual tenemos el COMSOAL de Arcus (1966, Pág. 259-278), el cual será uno de los temas centrales de este trabajo.

[12] https://upcommons.upc.edu/pfc/bitstream/2099.1/2587/1/31026-1.pdf,noviembre 2008

10.4 COMSOAL[13]

(Computer Method for sequencing operation for assembly lines).

En español sus siglas significan, Método Computarizado para la secuencia de operaciones en la línea de ensamble.

El COMSOAL es una heurística creada para dar solución al problema de equilibrado de líneas, pero también puede abordar otros tipos de problemas como en el caso de problemas de optimización combinatoria, donde se crea una lista de tareas para ser programadas, pero solo aquellas que son factibles, de lo contrario no aparecerán en la lista, de allí se escoge aleatoriamente la tarea a asignar. Es por eso que este método permite obtener soluciones de muy buena calidad.

Como se trabaja

El COMSOAL consta de los siguientes seis pasos:

PASO 1: Para cada tarea se debe identificar las tareas que le siguen en orden de precedencia.

PASO 2: Crear una LISTA A, que consiste en colocarle a cada tarea de la línea de ensamble, el numero de tareas que le preceden.

PASO 3: De la lista A, se crea una lista B, con las tareas que tienen cero predecesores, si no hay tareas por asignar entonces se debe parar.

PASO 4: De la lista B, se crea la lista C, compuesta por aquellas tareas cuyos tiempos de proceso no sea mayor al tiempo de ciclo disponible en la estación. Si esta lista esta vacía se debe abrir una nueva estación, la cual tendrá nuevamente todo el tiempo de ciclo disponible y se repite el paso 4.

PASO 5: De forma aleatoria se escoge de la lista C la tarea a asignarse a la estación.

PASO 6: Se debe actualizar el tiempo disponible en la estación y la lista B, con el fin de mirar el tiempo consumido y los predecesores completados hasta el momento. Si la lista B esta vacía se debe actualizar la lista A, y se vuelve al paso 3, de lo contrario se debe regresar al paso 4

[13] http://www.icaen.uiowa.edu/~dbricker/Stacks_pdf2/ALB_COMSOAL.pdf.noviembre 2008

Y su procedimiento es el siguiente:

1. Set x =0, UB= ∞, c= tiempo de ciclo
2. Empezar una nueva secuencia:

   Set x =x+1,

   A=TK

   NIPW (i) =NIP (i)
3. Precedencia factible:

   For all i en A, if NIPW (i)=0, add i to B
4. Tiempo viable:

   For all i en B, if $t_i \leq c$

   Add i to C

   If C esta vacía:

   Go to step 5, otherwise go to step 6
5. Abrir una nueva estación:

   IDLE=IDLE+c

   If IDLE>UB:

   Go to step 2, otherwise go to step 3
6. Seleccionar una tarea:

   Set m= card {C}

   Randomly generate RN Є U (0, 1)

   Let i^*= [m*RN] th task de C

   Remover i^* de A, B, C

   $c = c - t_i^*$

   For all i Є WIP (i^*)

   NIPW (i) = NIPW (i)-1

   If A is empty:

Go to step 7, otherwise go to step 3

7. Lista completa:

IDLE=IDLE+c

If IDLE≤UB, UB=IDLE and store Schedule

If x=X, stop

Otherwise go to step 2

Donde:

NIP (i)=NIPW (i): Numero de predecesores inmediatos para cada tarea i.

WIP (i): Indican para cuales otras tareas i tienen un inmediato predecesor.

TK: Consiste de N tareas.

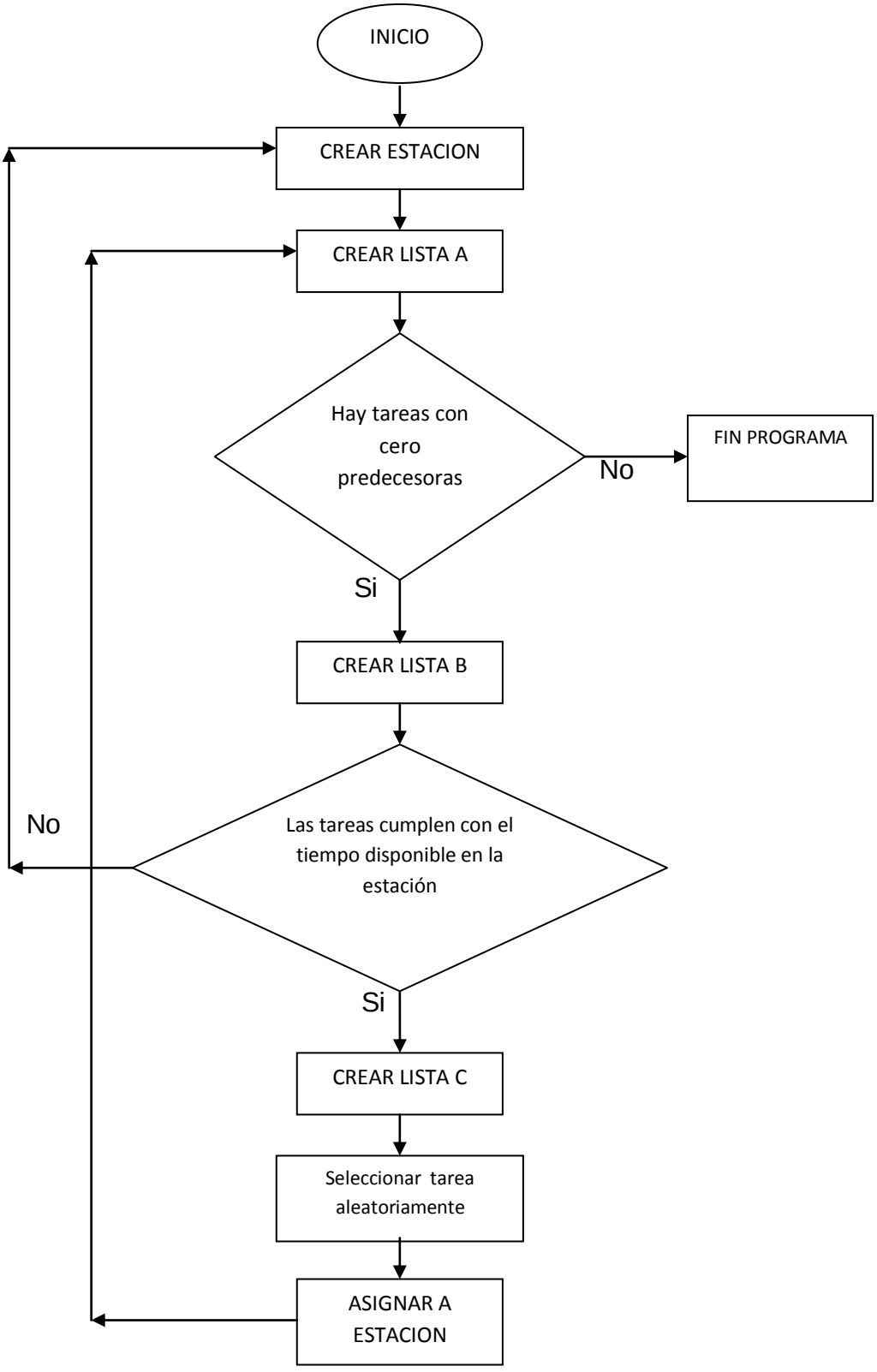
Durante cada generación de secuencia, estas listas son actualizadas.

A: Lista de tareas no asignadas.
B: tareas de A con todos los predecesores inmediatos.
C: Tareas de B con tiempo de operación que no excede el tiempo de ciclo en la estación de trabajo.

VENTAJAS

- Permite examinar un número grande de secuencias con un simple registro encontrando soluciones factibles y en poco tiempo.
- Es una técnica fácil de programar.
- El método solo tiene en cuenta aquellas tareas que cumplen con todas las restricciones en cada paso.
- Una secuencia es descartada cuando excede el límite superior.
- Una secuencia es guardada cuando se mejora el límite superior anterior.
- Las secuencias son generadas al escoger aleatoriamente una tarea y construye subsecuentes tareas.
- Nuevas estaciones son abiertas cuando se necesitan.

Figura 7. Flujograma Comsoal

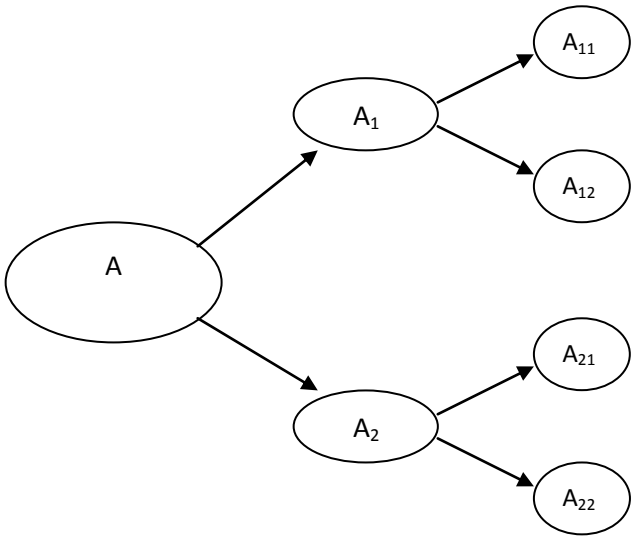


10.5 BRANCH AND BOUND (Ramificación y Acotamiento)

Técnica conocida por sus aplicaciones a los problemas de programación entera, la técnica se fundamenta en la idea de dividir y vencerás.
Para hacer el problema inicial o principal este se divide en subproblemas cada vez más pequeños.

División o ramificación

Figura 8. División o Ramificacion Branch and Bound



Se hace mediante una partición del conjunto completo de soluciones factibles en subconjuntos más pequeños

El sondeo o acotamiento

El sondeo o conquista básicamente es el acotamiento de la mejor solución en el subconjunto y después eliminando los subconjuntos cuya cota muestre que no es la indicada, pues no es posible que se obtenga una solución optima de dicha cota para el problema principal.

Existen tres formas de sondear:

1 Prueba: su cota <= Z *

2 Prueba: su soltura de PL no tiene soluciones factibles

3 Prueba : la solución optima para su soltura de PL es entera (si esta solución es mejor que la de apoyo , se convierte en la nueva solución de apoyo y se aplica de nuevo la prueba 1 a todos los subproblemas no sondeados, con la nueva Z * mejor)

Prueba de optimalidad

El problema termina cuando no existen subproblemas restantes o la solución de apoyo o incumbente actual es óptima. Si no es así se realiza otra iteración. (Si no existe una solución de apoyo o incumbente, la conclusión es que el problema no tiene soluciones factibles).

En resumen

Pasos iteración

1 Ramificación:
Entre los subproblemas (no sondeados) se elige el de creación mas reciente, se ramifica el nodo en dos subproblemas fijando la variable de ramificación.

2 Acotamiento:
Para cada nuevo subproblema se obtiene su cota aplicando por ejemplo el método simplex u otro método pertinente a su soltura de PL y redondeando hacia abajo el valor de Z en la solución optima.

3 Sondeo:
Para cada nuevo subproblema se aplican las tres pruebas de sondeo y se descartan aquellos problemas que quedan sondeados o eliminados por cualquiera de las tres pruebas.

Ejemplo 1

Supóngase que se tiene el siguiente problema de programación lineal entera:

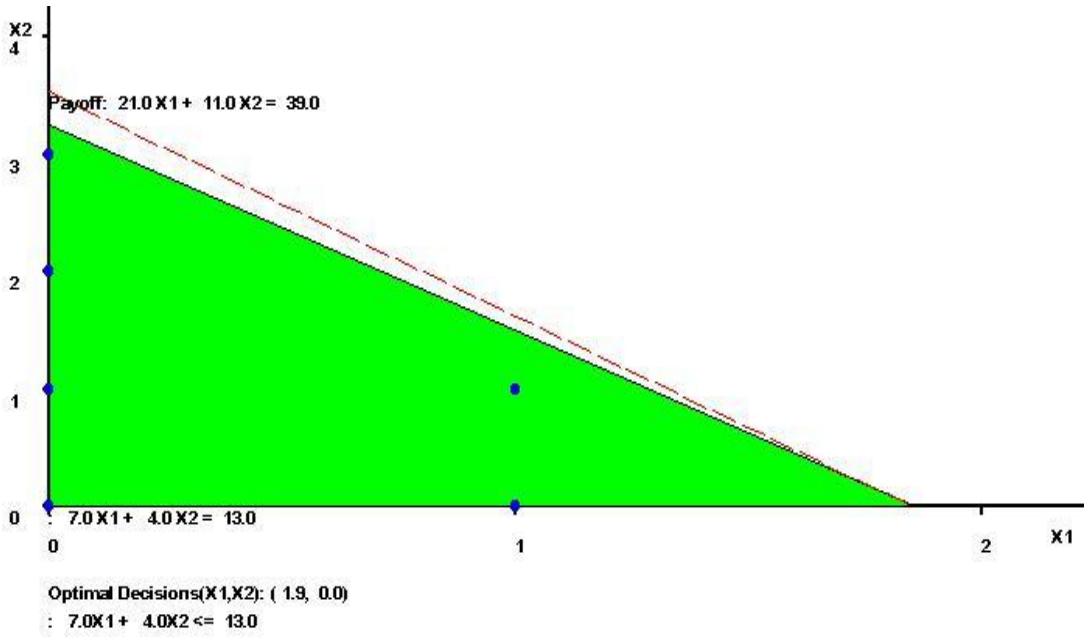
$$
\begin{aligned}
\text{PLE)} \quad \text{Max} \quad & 21x_1 + 11x_2 \\
\text{s.a.} \quad & 7x_1 + 4x_2 \leq 13 \\
& x_1 \geq 0 \\
& x_2 \geq 0 \\
& x_1, x_2 \text{ enteros}
\end{aligned}
$$

Se debe encontrar una solución óptima del problema, una de las formas es a través de la relajación continua, que consiste en encontrar una solución que acepta valores continuos, para el caso que se este maximizando corresponde al máximo valor que puede tomar la función objetivo, esta solución se encuentra por medio de diferentes algoritmos tanto exactos, heurísticas como

meta heurísticas y en el caso mas sencillo, el cual aplica para este ejemplo es el método grafico, debido a que posee únicamente dos variables y es mas practico.

Solución grafica

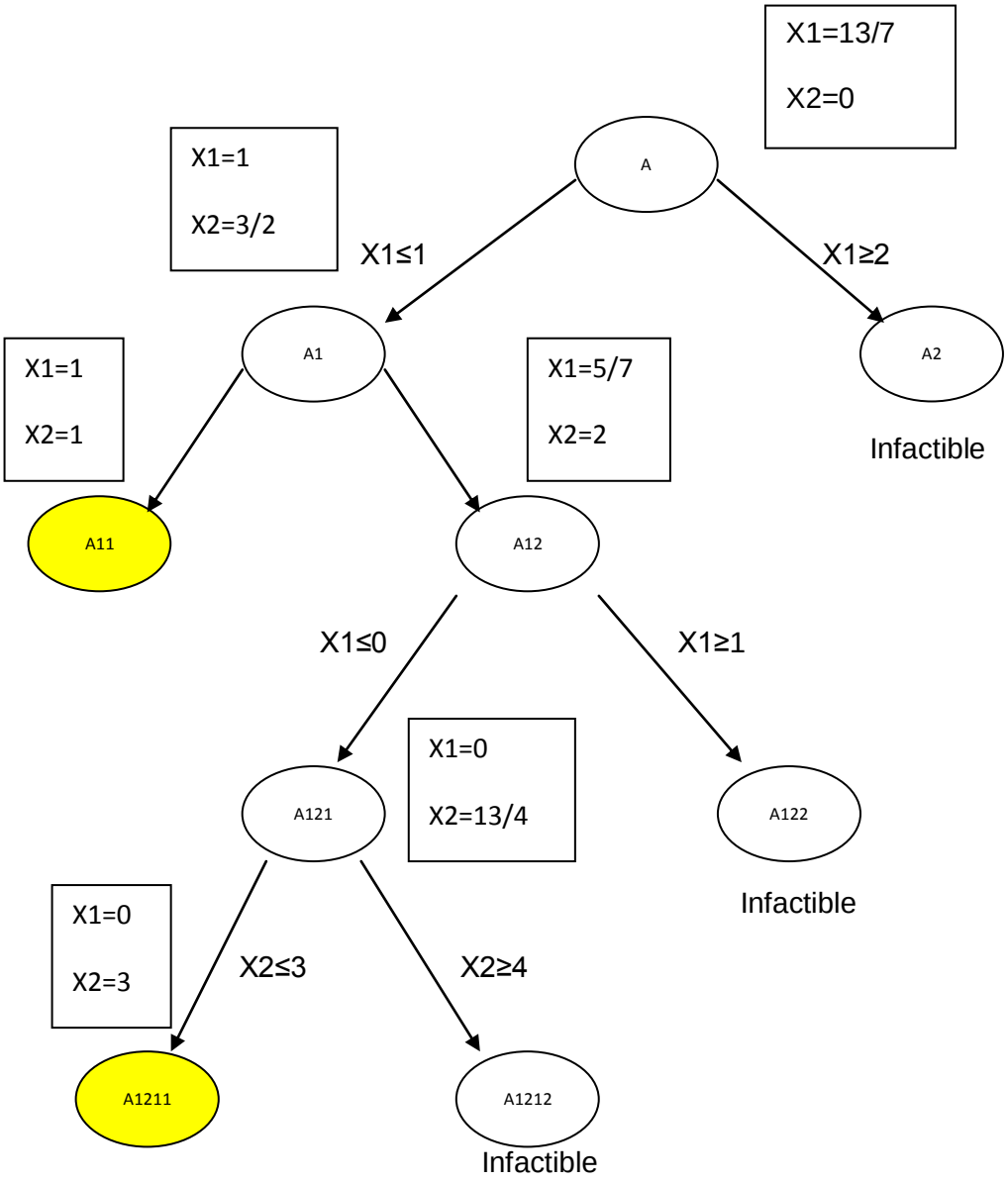
Figura 9. Método Grafico



La solución para relajación continua es X_1=1.9 y X_2=0, que cumple con las restricciones enunciadas en el problema, pero no cumplen con el problema de programación entera, pues el valor de X_1 no es entero.

Por tal motivo se debe utilizar otra estrategia para obtener la respuesta con valores enteros, en este caso el Branch and Bound es un método idóneo para la resolución de dicho problema.

Figura 10. Ramificación del ejemplo 1



A: corresponde a la relajación continua, hallada de forma gràfica.

A1: A+x1≤1, este subconjunto se obtiene aproximando el valor de x1 al entero inferior, en este caso a 1 y x2 se logra reemplazando x1 en la restricción.

A2: A + x1>=2, se aproxima la solución inicial X1=1,9 aproximada al entero superior. Es infactible porque no satisface la restricción.

A11: A1 + x2<=1 la solución óptima X1=1 y X2=1. Valor Óptimo Z=33. Debido a que la solución satisface las restricciones de integralidad, se termina este nodo.

A12: A1 + x2>=2 la solución X1=5/7 y X2=2. No es solución óptima de PLE debido a que X1 es aún fraccionario. Se continua el método debido a que el Valor Óptimo Z=37 es mayor que el Valor Óptimo de A11, en caso contrario se detiene el método y A11 sería la solución óptima de PLE.

A121: A12 + x1<=0 (X1=0 y X2=13/4. Z=35,75). Se continúa siguiendo el mismo razonamiento anterior

A122: A12 + X1>=1 Infactible.

A1211: A121 + X2<=3 (X1=0 y X2=3. Z=33) el Valor Óptimo más alto obtenido para los nodos con soluciones enteras. Se agota este nodo.

A1212: A121 + x2>=4 Infactible.

Por tanto, la solución óptima del problema de PLE es el nodo A1211, porque fue el que obtuvo el mayor valor de z con valores de las variables enteras.

11. DESARROLLO DE LA INVESTIGACION

11.1 PLANTEAMIENTO DEL PROBLEMA

En una empresa X, se tiene el siguiente diagrama de precedencia, correspondiente a la elaboración de un producto Y, y se requiere realizar un balanceo de línea que cumpla con los siguientes objetivos:

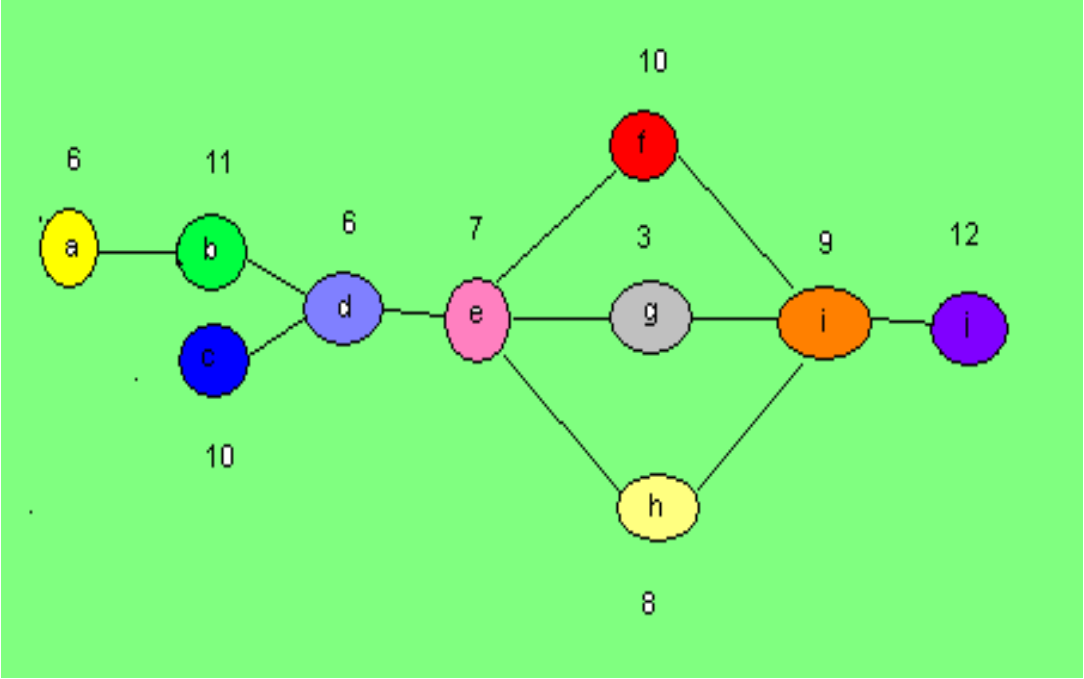
- Minimizar el número de estaciones, dado un tiempo de ciclo[14].
- Minimizar el tiempo de ciclo, dado un número determinado de estaciones[15].

Para ello se hará uso de dos técnicas o metodologías, ya mencionadas anteriormente, las cuales son COMSOAL y BRANCH AND BOUND (con Winqsb).

[14] Tiempo de ciclo= (tiempo de producción por día/producción por día en unidades), este tiempo depende de la demanda existente en el momento.

[15] Numero de estaciones teóricas=$\sum t_i/C$

Figura 11. Diagrama de Precedencia



MODELACION DEL SALBP-1

Función objetivo:

MIN Z: $Y_1+Y_2+Y_3+Y_4+Y_5$

Sujeto a:

Restricción 1

$X_{a,1}+X_{a,2}+X_{a,3}+X_{a,4}+X_{a,5}=1$

$X_{b,1}+X_{b,2}+X_{b,3}+X_{b,4}+X_{b,5}=1$

$X_{c,1}+X_{c,2}+X_{c,3}+X_{c,4}+X_{c,5}=1$

$X_{d,1}+X_{d,2}+X_{d,3}+X_{d,4}+X_{d,5}=1$

. .

. .

. .

$X_{i,1}+X_{i,2}+X_{i,3}+X_{i,4}+X_{i,5}=1$

$X_{j,1}+X_{j,2}+X_{j,3}+X_{j,4}+X_{j,5}=1$

Restricción 2

$6X_{a,1}+ 11X_{b,1} +10X_{c,1} +6 X_{d,1}+ 7X_{e,1} +10X_{f,1} +3X_{g,1}+ 8X_{h,1} +9X_{i,1} +12X_{j,1} \leq 21\ Y_1$

$6X_{a,2}+ 11X_{b,2} +10X_{c,2} +6 X_{d,2}+ 7X_{e,2} +10X_{f,2} +3X_{g,2}+ 8X_{h,2} +9X_{i,2} +12X_{j,2} \leq 21\ Y_2$

$6X_{a,3}+ 11X_{b,3} +10X_{c,3} +6 X_{d,3}+ 7X_{e,3} +10X_{f,3} +3X_{g,3}+ 8X_{h,3} +9X_{i,3} +12X_{j,3} \leq 21\ Y_1$

$6X_{a,4}+ 11X_{b,4} +10X_{c,4} +6 X_{d,4}+ 7X_{e,4} +10X_{f,4} +3X_{g,4}+ 8X_{h,4} +9X_{i,4} +12X_{j,4} \leq 21\ Y_4$

$6X_{a,5}+ 11X_{b,5} +10X_{c,5} +6 X_{d,5}+ 7X_{e,5} +10X_{f,5} +3X_{g,5}+ 8X_{h,5} +9X_{i,5} +12X_{j,5} \leq 21\ Y_5$

Restricción 3

$1X_{a,1}+ 2X_{a,2} +3X_{a,3} +4X_{a,4} +5X_{a,5} \leq 1X_{b,1}+ 2X_{b,2} +3X_{b,3} +4X_{b,4} +5X_{b,5}$

$1X_{b,1}+ 2X_{b,2} +3X_{b,3} +4X_{b,4} +5X_{b,5} \leq 1X_{d,1}+ 2X_{d,2} +3X_{d,3} +4X_{d,4} +5X_{d,5}$

$1X_{c,1}+ 2X_{c,2} +3X_{c,3} +4X_{c,4} +5X_{c,5} \leq 1X_{d,1}+ 2X_{d,2} +3X_{d,3} +4X_{d,4} +5X_{d,5}$

$1X_{d,1}+ 2X_{d,2} +3X_{d,3} +4X_{d,4} +5X_{d,5} \leq 1X_{e,1}+ 2X_{e,2} +3X_{e,3} +4X_{e,4} +5X_{e,5}$

$1X_{e,1}+ 2X_{e,2} +3X_{e,3} +4X_{e,4} +5X_{e,5} \leq 1X_{f,1}+ 2X_{f,2} +3X_{f,3} +4X_{f,4} +5X_{f,5}$

$1X_{e,1}+ 2X_{e,2} +3X_{e,3} +4X_{e,4} +5X_{e,5} \leq 1X_{g,1}+ 2X_{g,2} +3X_{g,3} +4X_{g,4} +5X_{g,5}$

$1X_{e,1}+ 2X_{e,2} +3X_{e,3} +4X_{e,4} +5X_{e,5} \leq 1X_{h,1}+ 2X_{h,2} +3X_{h,3} +4X_{h,4} +5X_{h,5}$

$1X_{f,1}+ 2X_{f,2} +3X_{f,3} +4X_{f,4} + 5X_{f,5} \leq 1X_{i,1}+ 2X_{i,2} +3X_{i,3} +4X_{i,4} +5X_{i,5}$

$1X_{g,1}+ 2X_{g,2} +3X_{g,3} +4X_{g,4} +5X_{g,5} \leq 1X_{i,1}+ 2X_{i,2} +3X_{i,3} +4X_{i,4} +5X_{i,5}$

$1X_{h,1}+ 2X_{h,2} +3X_{h,3} +4X_{h,4} +5X_{h,5} \leq 1X_{i,1}+ 2X_{i,2} +3X_{i,3} +4X_{i,4} +5X_{i,5}$

$1X_{i,1}+ 2X_{i,2} +3X_{i,3} +4X_{i,4} +5X_{i,5} \leq 1X_{j,1}+ 2X_{j,2} +3X_{j,3} +4X_{j,4} +5X_{j,5}$

Restricción 4

$Y_2 \leq Y_1$

$Y_3 \leq Y_2$

$Y_4 \leq Y_3$

$Y_5 \leq Y_4$

MODELACION DEL SALBP-2

Función objetivo:

MIN Z: C

Sujeto a:

Restricción 1

$X_{a,1} + X_{a,2} + X_{a,3} + X_{a,4} + X_{a,5} = 1$

$X_{b,1} + X_{b,2} + X_{b,3} + X_{b,4} + X_{b,5} = 1$

$X_{c,1} + X_{c,2} + X_{c,3} + X_{c,4} + X_{c,5} = 1$

$X_{d,1} + X_{d,2} + X_{d,3} + X_{d,4} + X_{d,5} = 1$

. .

. .

$X_{i,1} + X_{i,2} + X_{i,3} + X_{i,4} + X_{i,5} = 1$

$X_{j,1} + X_{j,2} + X_{j,3} + X_{j,4} + X_{j,5} = 1$

Restricción 2

$6X_{a,1} + 11X_{b,1} + 10X_{c,1} + 6X_{d,1} + 7X_{e,1} + 10X_{f,1} + 3X_{g,1} + 8X_{h,1} + 9X_{i,1} + 12X_{j,1} \leq C$

$6X_{a,2} + 11X_{b,2} + 10X_{c,2} + 6X_{d,2} + 7X_{e,2} + 10X_{f,2} + 3X_{g,2} + 8X_{h,2} + 9X_{i,2} + 12X_{j,2} \leq C$

$6X_{a,3} + 11X_{b,3} + 10X_{c,3} + 6X_{d,3} + 7X_{e,3} + 10X_{f,3} + 3X_{g,3} + 8X_{h,3} + 9X_{i,3} + 12X_{j,3} \leq C$

$6X_{a,4} + 11X_{b,4} + 10X_{c,4} + 6X_{d,4} + 7X_{e,4} + 10X_{f,4} + 3X_{g,4} + 8X_{h,4} + 9X_{i,4} + 12X_{j,4} \leq C$

$6X_{a,5} + 11X_{b,5} + 10X_{c,5} + 6X_{d,5} + 7X_{e,5} + 10X_{f,5} + 3X_{g,5} + 8X_{h,5} + 9X_{i,5} + 12X_{j,5} \leq C$

Restricción 3

$1X_{a,1} + 2X_{a,2} + 3X_{a,3} + 4X_{a,4} + 5X_{a,5} \leq 1X_{b,1} + 2X_{b,2} + 3X_{b,3} + 4X_{b,4} + 5X_{b,5}$

$1X_{b,1} + 2X_{b,2} + 3X_{b,3} + 4X_{b,4} + 5X_{b,5} \leq 1X_{d,1} + 2X_{d,2} + 3X_{d,3} + 4X_{d,4} + 5X_{d,5}$

$1X_{c,1} + 2X_{c,2} + 3X_{c,3} + 4X_{c,4} + 5X_{c,5} \leq 1X_{d,1} + 2X_{d,2} + 3X_{d,3} + 4X_{d,4} + 5X_{d,5}$

$1X_{d,1} + 2X_{d,2} + 3X_{d,3} + 4X_{d,4} + 5X_{d,5} \leq 1X_{e,1} + 2X_{e,2} + 3X_{e,3} + 4X_{e,4} + 5X_{e,5}$

$1X_{e,1} + 2X_{e,2} + 3X_{e,3} + 4X_{e,4} + 5X_{e,5} \leq 1X_{f,1} + 2X_{f,2} + 3X_{f,3} + 4X_{f,4} + 5X_{f,5}$

$1X_{e,1} + 2X_{e,2} + 3X_{e,3} + 4X_{e,4} + 5X_{e,5} \leq 1X_{g,1} + 2X_{g,2} + 3X_{g,3} + 4X_{g,4} + 5X_{g,5}$

$$1X_{e,1} + 2X_{e,2} + 3X_{e,3} + 4X_{e,4} + 5X_{e,5} \leq 1X_{h,1} + 2X_{h,2} + 3X_{h,3} + 4X_{h,4} + 5X_{h,5}$$

$$1X_{f,1} + 2X_{f,2} + 3X_{f,3} + 4X_{f,4} + 5X_{f,5} \leq 1X_{i,1} + 2X_{i,2} + 3X_{i,3} + 4X_{i,4} + 5X_{i,5}$$

$$1X_{g,1} + 2X_{g,2} + 3X_{g,3} + 4X_{g,4} + 5X_{g,5} \leq 1X_{i,1} + 2X_{i,2} + 3X_{i,3} + 4X_{i,4} + 5X_{i,5}$$

$$1X_{h,1} + 2X_{h,2} + 3X_{h,3} + 4X_{h,4} + 5X_{h,5} \leq 1X_{i,1} + 2X_{i,2} + 3X_{i,3} + 4X_{i,4} + 5X_{i,5}$$

$$1X_{i,1} + 2X_{i,2} + 3X_{i,3} + 4X_{i,4} + 5X_{i,5} \leq 1X_{j,1} + 2X_{j,2} + 3X_{j,3} + 4X_{j,4} + 5X_{j,5}$$

11.1.1 RESOLUCION SALBP-1 CON COMSOAL

El SALBP-1 se utiliza en casos donde se esta iniciando un proceso productivo y la demanda es conocida, por lo que es necesario determinar el número mínimo de estaciones para cumplir con la producción y al mismo tiempo reducir costos de montaje de planta.
En este tipo de problema el tiempo de ciclo es conocido, para el caso de la empresa X, el tiempo es de 21 y se requiere encontrar una distribución de las tareas que permitan un balanceo óptimo, utilizando un mínimo de estaciones de trabajo.

ITERACION 1

C=21
∑ti=82
Estaciones teóricas[16]= $\sum t_i/c=4$

Se crea una lista A en donde se ubican las tareas, tiempos de las tareas, tareas predecesoras y numero total de tareas predecesoras así:

Tabla 1 Lista A 1.1.1

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
b	11	a	1
c	10	-	0
d	6	b-c	2
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	I	1

[16] Es el número mínimo teórico que se puede encontrar de estaciones de trabajo, pero que en ocasiones es difícil obtener.

Se crea una lista B con aquellas tareas que no tienen predecesor en la lista A así:
Tarea a y tarea c cumplen. (Ver Figura 11)

Tabla 2 Lista B 1.1.1

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

Se crea una lista C en la cual se agrupan aquellas tareas que cumplen con el tiempo disponible en la estación (en este caso tiempo de ciclo = 21) así:

Tabla 3 Lista C 1.1.1

Td= 21

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

Tarea a (tj=6) y tarea c (tj=10) ≤ 21. Como las dos cumplen se debe escoger una de manera aleatoria, para este ejemplo aplicamos shift Ran y se obtuvo 0,507 por lo que se escoge la tarea c porque su rango de probabilidades se encuentra entre 0,5 y 1. La tarea c tiene asignado un tiempo de 10, el cual se resta del tiempo disponible (21), y queda como tiempo disponible para la estación 1: (11). Si la próxima tarea o tareas que resulten de aplicar de nuevo el paso 2 no cumplen con la restricción de tiempo se debe crear una nueva estación, como se vera mas adelante en el ejemplo.

La tarea c queda entonces asignada a la estación numero 1.

Se crea nuevamente una lista A1.1 sin la estación c y recordando que como tiempo disponible queda (11).Se repite todo el proceso hasta que todas las tareas queden asignadas a una estación.

Tabla 4 Lista A 1.1.2

TAREA	Tj	PREDEC.	# PREDEC.
a	6	-	0
b	11	a	1
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 5 Lista B 1.1.2

TAREA	Tj	PREDEC.	# PREDEC.
a	6	-	0

Tabla 6 Lista C 1.1.2

Td= 11

TAREA	Tj	PREDEC.	# PREDEC.
a	6	-	0

Tarea a tj =6 ≤11 (tiempo disponible) .Cumple con la restricción de tiempo por lo tanto la tarea a queda en la estación 1 en donde también esta asignada la tarea c. Entre las dos se genera un tiempo acumulado de (16), por lo tanto como disponible queda (5).

Tabla 7 Lista A 1.1.3

TAREA	Tj	PREDEC.	# PREDEC.
b	11	-	0
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 8 Lista B 1.1.3

TAREA	Tj	PREDEC.	# PREDEC.
b	11	-	0

Tarea b tj=11 >5 que es el tiempo disponible en la estación 1 después de asignar las tareas a y c. Como el tiempo de la tarea b es mayor al disponible se debe crear una nueva estación, la estación 2. Se continúa con el algoritmo y se siguen asignando las tareas a la estación 2 cumpliendo con las restricciones de tiempo, cada vez que se abre una estación nueva se inicia con un tiempo disponible de (21) que es el tiempo de ciclo para el ejemplo dado.

Tabla 9 Lista A 1.1.4

TAREA	Tj	PREDEC.	# PREDEC.
d	6	-	0
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 10 Lista B 1.1.4

TAREA	Tj	PREDEC.	# PREDEC.
d	6	-	0

Tabla 11 Lista C 1.1.3

Td= 10

TAREA	Tj	PREDEC.	# PREDEC.
d	6	-	0

Tarea d tj=6 ≤ 10 .Cumple con el tiempo por lo tanto queda en la estación 2.

Tabla 12 Lista A 1.1.5

TAREA	Tj	PREDEC.	# PREDEC.
e	7	-	0
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 13 Lista B 1.1.5

TAREA	Tj	PREDEC.	# PREDEC.
e	7	-	0

Tarea e tj=7 > 4 No cumple, por lo tanto se debe abrir otra estación, la estación 3.

Tabla 14 Lista A 1.1.6

TAREA	Tj	PREDEC.	# PREDEC.
h	8	-	0
g	3	-	0
f	10	-	0
i	9	h-f-g	3
j	12	i	1

Tabla 15 Lista B 1.1.6

TAREA	Tj	PREDEC.	# PREDEC.
h	8	-	0
g	3	-	0
f	10	-	0

Tabla 16 Lista C 1.1.4

Td= 14

TAREA	Tj	PREDEC.	# PREDEC.
h	8	-	0
g	3	-	0
f	10	-	0

Tarea h tj=8, Tarea g tj=3, Tarea f tj=10. ≤ 14. Todas cumplen por lo tanto se escoge de manera aleatoria, empleando nuevamente shift Ran se obtiene 0,5 y se escoge la tarea g.

Tabla 17 Lista A 1.1.7

TAREA	Tj	PREDEC.	# PREDEC.
h	8	-	0
f	10	-	0
i	9	h-f	2
j	12	i	1

Tabla 18 Lista B 1.1.7

TAREA	Tj	PREDEC.	# PREDEC.
h	8	-	0
f	10	-	0

Tabla 19 Lista C 1.1.5

Td= 11

TAREA	Tj	PREDEC.	# PREDEC.
h	8	-	0
f	10	-	0

Tarea h tj=8 Tarea f tj=10. ≤ 11 Como las dos cumplen, nuevamente se escoge de manera aleatoria y se determina que la tarea h queda en la estación 3.

Tabla 20 Lista A 1.1.8

TAREA	Tj	PREDEC.	# PREDEC.
f	10	-	0
i	9	f	1
j	12	i	1

Tabla 21 Lista B 1.1.8

TAREA	Tj	PREDEC.	# PREDEC.
f	10	-	0

Tarea f tj=10 > 3 (Tiempo disponible) Como no cumple se abre una nueva estación, la estación 4

Tabla 22 Lista A 1.1.9

TAREA	Tj	PREDEC.	# PREDEC.
i	9	-	0
j	12	i	1

Tabla 23 Lista B 1.1.9

TAREA	Tj	PREDEC.	# PREDEC.
i	9	-	0

Tabla 24 Lista C 1.1.6 Td= 11

TAREA	Tj	PREDEC.	# PREDEC.
i	9	-	0

Tarea i tj=9 ≤ 11 Cumple.

Tabla 25 Lista A 1.1.10

TAREA	Tj	PREDEC.	# PREDEC.
j	12	-	0

Tabla 26 Lista B 1.1.10

TAREA	Tj	PREDEC.	# PREDEC.
j	12	-	0

Tarea j tj=12 > 2 (Tiempo disponible) Como no cumple se abre una nueva estación, la estación 5.

Las estaciones quedan de la siguiente manera:

Tabla 27 Estaciones iteración 1 SALBP-1

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	c	10	10	5
	a	6	16	
2	b	11	11	4
	d	6	17	
3	e	7	7	3
	g	3	10	
	h	8	18	
4	f	10	10	2
	i	9	19	
5	j	12	12	9

EFICIENCIA:

E= ∑Tj / (m* c) = 82 / (5* 19)= 86.3%

TIEMPO MUERTO TOTAL: 23

Como resultado se obtuvieron más de 4 estaciones por lo tanto repetimos el algoritmo iniciando por otra ruta.

ITERACION 2 SALBP – 1

Tabla 28 Lista A 1.2.1

TAREA	Tj.	PREDECE.	# PREDECE.
a	6	-	0
b	11	a	1
c	10	-	0
d	6	b-c	2
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 29 Lista B 1.2.1

TAREA	Tj.	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

Tabla 30 Lista C 1.2.1

Td = 21

TAREA	Tj.	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

Tarea a tj=6 Tarea c tj=10 ≤ 21 Pero para este caso se inicia por una nueva ruta, ahora se escoge la tarea a para la estación número 1.

Tabla 31 Lista A 1.2.2

TAREA	Tj	PREDEC.	# PREDEC.
b	11	-	0
c	10	-	0
d	6	b-c	2
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 32 Lista B 1.2.2

TAREA	Tj	PREDEC.	# PREDEC.
b	11	-	0
c	10	-	0

Tabla 33 Lista C 1.2.2

Td = 15

TAREA	Tj	PREDEC.	# PREDEC.
b	11	-	0
c	10	-	0

Tarea c tj=10, Tarea b tj=11 ≤ 15 Las dos cumplen, por lo tanto se escoge una de manera aleatoria shift Ran 0,622 y la tarea b se asigna a la estación 1

Tabla 34 Lista A 1.2.3

TAREA	Tj.	PREDEC.	# PREDEC.
c	10	-	0
d	6	c	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	f-g-h	3
j	12	i	1

Tabla 35 Lista B 1.2.3

TAREA	Tj.	PREDEC.	# PREDEC.
c	10	-	0

Tarea c tj=10 > 4 (Tiempo disponible) Como no cumple se abre una nueva estación para esta tarea, la estación 2.

Tabla 36 Lista A 1.2.4

TAREA	Tj.	PREDEC.	#PREDEC.
d	6	-	0
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	f-g-h	3
j	12	i	1

Tabla 37 Lista B 1.2.4

TAREA	Tj.	PREDEC.	#PREDEC.
d	6	-	0

Tabla 38 Lista C 1.2.3

Td= 11

TAREA	Tj.	PREDEC.	#PREDEC.
d	6	-	0

Tarea d tj=6 ≤ 11 La tarea d es asignada a la estación numero 2

Tabla 39 Lista A 1.2.5

TAREA	Tj.	PREDEC.	# PREDEC.
e	7	-	0
h	8	e	1
g	3	e	1
f	10	e	1
i	9	f-g-h	3
j	12	i	1

Tabla 40 Lista B 1.2.5

TAREA	Tj.	PREDEC.	# PREDEC.
e	7	-	0

Tarea e tj=7 > 5 No cumple por lo tanto se abre una nueva estación para la tarea e, la estación 3.

Tabla 41 Lista A 1.2.6

TAREA	Tj.	PREDEC.	# PREDEC.
h	8	-	0
g	3	-	0
f	10	-	0
i	9	f-g-h	3
j	12	i	1

Tabla 42 Lista B 1.2.6

TAREA	Tj.	PREDEC.	# PREDEC.
h	8	-	0
g	3	-	0
f	10	-	0

Tabla 43 Lista C 1.2.4

Td = 14

TAREA	Tj.	PREDEC.	# PREDEC.
h	8	-	0
g	3	-	0
f	10	-	0

Tarea h tj=8, Tarea g tj=3, Tarea f tj=10. ≤ 14 Todas cumplen, entonces de manera aleatoria 0,29 y la tarea h se designa a la estación numero 3.

Tabla 44 Lista A 1.2.7

TAREA	Tj.	PREDEC.	# PREDEC.
g	3	-	0
f	10	-	0
i	9	g-f	2
j	12	i	1

Tabla 45 Lista B 1.2.7

TAREA	Tj.	PREDEC.	# PREDEC.
g	3	-	0
f	10	-	0

Tabla 46 Lista C 1.2.5

Td = 6

TAREA	Tj.	PREDEC.	# PREDEC.
g	3	-	0

Tarea g tj=3 ≤ 6 La tarea g queda asignada a la estación numero 3

Tabla 47 Lista A 1.2.8

TAREA	Tj.	PREDEC.	# PREDEC.
f	10	-	0
i	9	f	1
j	12	i	1

Tabla 48 Lista B 1.2.8

TAREA	Tj.	PREDEC.	# PREDEC.
f	10	-	0

Tarea f tj=10 > 3. Por lo tanto se debe abrir una nueva estación para la tarea f, la estación numero 4.

Tabla 49 Lista A 1.2.9

TAREA	Tj.	PREDEC.	# PREDEC.
i	9	-	0
j	12	i	1

Tabla 50 Lista B 1.2.9

TAREA	Tj.	PREDEC.	# PREDEC.
i	9	-	0

Tabla 51 Lista C 1.2.6

Td = 11

TAREA	Tj.	PREDEC.	# PREDEC.
i	9	-	0

Tarea i tj=9 ≤ 11 Por lo tanto la tarea i queda asignada a la estación numero 4.

Tabla 52 Lista A 1.2.10

TAREA	Tj.	PREDEC.	# PREDEC.
j	12	-	0

Tabla 53 Lista B 1.2.10

TAREA	Tj.	PREDEC.	# PREDEC.
j	12	-	0

Tarea j tj=12 > 2. (Tiempo disponible)

Para la tarea j se debe abrir una nueva estación, la estación numero 5 pues el tiempo disponible en la estación 4 es de 2 y la tarea j debe disponer de un tiempo mínimo de 12.

Las estaciones quedan de la siguiente manera:

Tabla 54 Estaciones iteración 2 SALB-1

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	a	6	6	4
	b	11	17	
2	c	10	10	5
	d	6	16	
3	e	7	7	3
	h	8	15	
	g	3	18	
4	f	10	10	2
	i	9	19	
5	j	12	12	9

.

El SALBP- 2 hace variar el tiempo de ciclo para un número

11.1.2 RESOLUCION SALBP-2 CON COMSOAL

A continuación veremos la aplicación del SALBP-2, pero hay que aclarar que examinando la teoría no existen métodos propios para resolverlo directamente, sino que el SALBP-2 se apoya del SALBP-1 para su resolución dado un número de estaciones de trabajo y lo resuelve como si fuera un SALBP-1, si el resultado arrojado da un numero mayor de estaciones de trabajo o si el tiempo de ciclo es peor que la anterior respuesta alcanzada, se descarta y se repite nuevamente el proceso hasta alcanzar un tiempo de ciclo mínimo con el numero de estaciones preestablecido, como se vera a continuación.

El SALBP-2, se utiliza para aquellos casos donde el proceso productivo ya existe, es decir las estaciones ya están creadas, por lo que ahora se busca es mejorar la productividad mejorando el tiempo de ciclo, o sea, disminuyéndolo.

Los datos para este ejemplo son los siguientes:

ITERACION 1

$m^{17}=5$

$\sum t_i=82$

Se iniciara tomando un tiempo de ciclo de c=23, para ver que sucede.

Se crea la lista A con las tareas que no han sido asignadas, sus tiempos de operación y el número de predecesoras que tenga.

Tabla 55 Lista A 2.1.1

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
b	11	a	1
c	10	-	0
d	6	b-c	2
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

De la lista A, se origina la lista B con aquellas tareas que tienen cero predecesoras.

[17] El numero de estaciones(m), se tomo teniendo en cuenta los resultados del SALBP-1

Tabla 56 Lista B 2.1.1

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

De la lista B, se crea la lista C, con aquellas tareas que cumplen con el tiempo disponible en la estación (Tj<=Td), en este caso el tiempo disponible (Td) corresponde al tiempo de ciclo de 23.

Tabla 57 Lista C 2.1.1

Td=23

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

Como en la lista C existen dos tareas que cumplen con el requisito del tiempo disponible, se debe escoger una de ellas de forma aleatoria, en este caso se hará de una forma simple que es a través del Shift Ran que brinda la calculadora, en este caso arrojo el valor de 0,061, que equivale a la tarea a, cuyo valor de probabilidad esta entre 0 y 0,5.

Por tanto se asigna la tarea a, en la estación numero 1 y se debe calcular el tiempo disponible que queda en la estación de la siguiente forma:

El tiempo disponible= el tiempo disponible anterior – el tiempo de operación de la tarea asignada.

Td=23-6=17, y se repite nuevamente todo el proceso.

Se crea la lista A1 con las tareas que no han sido asignadas.

Tabla 58 Lista A 2.1.2

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0
c	10	-	0
d	6	b-c	2
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Se origina la lista B1, con aquellas tareas que tienen cero predecesoras.

Tabla 59 Lista B 2.1.2

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0
c	10	-	0

Se crea la lista C1, con aquellas tareas que cumplan con el tiempo disponible en la estación, el cual es de 17.

Tabla 60 Lista C 2.1.2 Td=17

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0
c	10	-	0

Como existen dos tareas que cumplen con el tiempo de ciclo, se escoge aleatoriamente una tarea a través de shift Ran, que arrojo un valor de 0,403, que equivale a la tarea b, la cual debe asignarse a la estación 1 y se calcula el tiempo disponible en la estación así: 17-11=6.

Nuevamente se crea la lista A.

Tabla 61 Lista A 2.1.3

TAREA	Tj	PREDECE.	# PREDECE.
c	10	-	0
d	6	c	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 62 Lista B 2.1.3

TAREA	Tj	PREDECE.	# PREDECE.
c	10	-	0

Tabla 63 Lista C 2.1.3

TAREA	Tj	PREDECE.	# PREDECE.
-	-	-	-

Como la lista C esta vacía se abre una nueva estación, con el tiempo de ciclo completo de 23.

Tabla 64 Lista A 2.1.4

TAREA	Tj	PREDECE.	# PREDECE.
c	10	-	0
d	6	c	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 65 Lista B 2.1.4

TAREA	Tj	PREDECE.	# PREDECE.
c	10	-	0

Tabla 66 Lista C 2.1.4

Td=23

TAREA	Tj	PREDECE.	# PREDECE.
c	10	-	0

Como solo existe la tarea c, se asigna a la estación 2 y se recalcula el tiempo disponible en la estación así: 23-10=13.

Actualizamos la lista A:

Tabla 67 Lista A 2.1.5

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	0
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 68 Lista B 2.1.5

TAREA	Tj	PREDECE.	# PREDECE.
e	6	-	0

Tabla 69 Lista C 2.1.5

Td=13

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	0

Como solo existe la tarea d en la lista C, se asigna a la estación 2, se actualiza la lista A y se calcula el tiempo disponible en la estación: 13-6=7

Tabla 70 Lista A 2.1.6

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 71 Lista B 2.1.6

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0

Tabla 72 Lista C 2.1.6

Td=7

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0

Se asigna la tarea e en la estación numero 2, debido a que cumple con el tiempo disponible en la estación y se calcula el nuevo tiempo disponible en la estación: 7-7=0, como no queda tiempo disponible se abre una nueva estación con un tiempo de 23 y se actualiza la lista A.

Tabla 73 Lista A 2.1.7

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0
i	9	h-f-g	3
j	12	i	1

Tabla 74 Lista B 2.1.7

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0

Tabla 75 Lista C 2.1.7

Td=23

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0

Como la lista C, consta de tres tareas se debe escoger una aleatoriamente a través del Shift Ran que dio un valor de 0,536, que equivale a la tarea g para ser asignada a la estación numero 3.

Se actualiza la lista A y se calcula el tiempo disponible en la estación: 23-3=20

Tabla 76 Lista A 2.1.8

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
f	10	-	0
i	9	h-f	2
j	12	i	1

Tabla 77 Lista B 2.1.8

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
f	10	-	0

Tabla 78 Lista C 2.1.8

Td=20

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
f	10	-	0

Se debe escoger aleatoriamente una tarea de la lista C, utilizando Shift Ran da un valor de 0,374, equivalente a la tarea h y se asigna a la estación 3.
Se calcula el tiempo disponible Td=20-8=12 y se actualiza la lista A.

Tabla 79 Lista A 2.1.9

TAREA	Tj	PREDECE.	# PREDECE.
f	10	-	0
i	9	f	1
j	12	i	1

Tabla 80 Lista B 2.1.9

TAREA	Tj	PREDECE.	# PREDECE.
f	10	-	0

Se crea la lista C, con aquellas tareas de la lista B que cumplan con el tiempo disponible de 12.

Tabla 81 Lista C 2.1.9

TAREA	Tj	PREDECE.	# PREDECE.
f	10	-	0

Como solo hay una tarea que cumple con el tiempo de la estación, se asigna la tarea f a la estación numero 3. Se calcula el tiempo disponible Td=12-10=2 y se actualiza la lista A.

Nota: observando el diagrama de precedencia, se nota que la tarea i y j tienen un tiempo de proceso mayor al tiempo disponible en la estación 3, por lo cual se debe abrir una nueva estación, la numero 4, con el tiempo completo de 23.

Tabla 82 Lista A 2.1.10

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0
j	12	i	1

Tabla 83 Lista B 2.1.10

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0

Tabla 84 Lista C 2.1.10

Td=23

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0

En la lista C, como solo existe la tarea i, se asigna a la estación numero 4, se calcula el tiempo disponible Td=23-9=14, y se actualiza la lista A.

Tabla 85 Lista A 2.1.11

TAREA	Tj	PREDECE.	# PREDECE.
j	12	-	0

Tabla 86 Lista B 2.1.11

TAREA	Tj	PREDECE.	# PREDECE.
j	12	-	0

Tabla 87 Lista C 2.1.11

TAREA	Tj	PREDECE.	# PREDECE.
j	12	-	0

Se asigna la tarea j a la estación numero 4.

Tabla 88 Estaciones iteración 1 SALB-2

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	a	6	6	6
	b	11	17	
2	c	10	10	0
	d	6	16	
	e	7	23	
3	g	3	3	2
	h	8	11	
	f	10	21	
4	i	9	9	2
	j	12	21	

La anterior iteración iniciando con un tiempo de ciclo de 23, arroja los siguientes resultados:

Tiempo de ciclo del balanceo

C=23

Numero de estaciones

m=4

Eficiencia del balanceo

E=∑ Ti/ (m*c)

E=82/(4*23)

E=89,1%

Tiempo muerto total o demora del balanceo

D= (m*c)- ∑ Ti

D= (4*23)-82

D= 10

Esta respuesta, brinda un tiempo de ciclo de 23 con 4 estaciones, la cual no alcanza el objetivo de minimizar el tiempo de ciclo dado un numero de estaciones de trabajo, por tal motivo no sirve para el caso SALBP 2, pero si

sirve para el caso SALBP 1, pues brinda un numero mínimo de estaciones de trabajo para un tiempo de ciclo dado, en este caso de 23.

En consecuencia como la respuesta no fue positiva, pues no se mejoro el tiempo de ciclo, se debe variar y ver si los resultados son mejores.

ITERACION 2

El número de estaciones sigue siendo de 5, pero el tiempo de ciclo, ahora será de 21.

Tabla 89 Lista A 2.2.1

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
b	11	a	1
c	10	-	0
d	6	b-c	2
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 90 Lista B 2.2.1

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

Tabla 91 Lista C 2.2.1

Td= C=21

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

A través del Shift Ran que arrojo un valor de 0,887, se elige la tarea c, para ser asignada a la estación numero 1. Se calcula el tiempo disponible en la estación Td=21-10=11 y se actualiza la lista A.

Tabla 92 Lista A 2.2.2

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
b	11	a	1
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 93 Lista B 2.2.2

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0

Tabla 94 Lista C 2.2.2

Td=11

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0

Se asigna la tarea a en la estación numero 1, se calcula el tiempo disponible Td=11-6=5 y se actualiza la lista A.

Tabla 95 Lista A 2.2.3

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 96 Lista B 2.2.3

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0

Tabla 97 Lista C 2.2.3

Td=5

TAREA	Tj	PREDECE.	# PREDECE.
-	-	-	-

Como la lista C esta vacía se debe abrir una nueva estación, con todo el tiempo disponible de 21.

Tabla 98 Lista A 2.2.4

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 99 Lista B 2.2.4

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0

Tabla 100 Lista C 2.2.4

Td=21

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0

Como solo esta la tarea b, se debe asignar a la estación 2, se calcula el tiempo disponible Td=21-11=10, y se actualiza la lista A.

Tabla 101 Lista A 2.2.5

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	0
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 102 Lista B 2.2.5

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	0

Tabla 103 Lista C 2.2.5

Td=10

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	0

Se asigna la tarea d a la estación 2, se calcula el tiempo disponible Td=10-6=4 y se actualiza la lista A.

Tabla 104 Lista A 2.2.6

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0
h	8	e	1

g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 105 Lista B 2.2.6

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0

Tabla 106 Lista C 2.2.6

Td=4

TAREA	Tj	PREDECE.	# PREDECE.
-	-	-	-

Se debe abrir una nueva estación, la numero 3, con todo el tiempo disponible de 21, debido a que la lista C esta vacía.

Tabla 107 Lista A 2.2.7

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 108 Lista B 2.2.7

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0

Tabla 109 Lista C 2.2.7

Td=21

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0

Se asigna la tarea e en la estación numero 3, se calcula el tiempo disponible Td=21-7=14 y se actualiza la lista A.

Tabla 110 Lista A 2.2.8

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0
i	9	h-f-g	3
j	12	i	1

Tabla 111 Lista B 2.2.8

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0

Tabla 112 Lista C 2.2.8

Td=14

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0

Como existe mas de una tarea, se debe escoger una aleatoriamente, a través del Shift Ran, que da un valor de 0,596, que equivale a escoger la tarea g para ser asignada a la estación numero 3, se actualiza el tiempo disponible Td=14-3=11 y se actualiza la lista A.

Tabla 113 Lista A 2.2.9

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
f	10	-	0
i	9	h-f	2
j	12	i	1

Tabla 114 Lista B 2.2.9

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
f	10	-	0

Tabla 115 Lista C 2.2.9

Td=11

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
f	10	-	0

Nuevamente se escoge de forma aleatoria a través del Shift Ran la tarea para ser asignada a la estación 3, brindando un valor de 0,727, que corresponde a la tarea f, se recalcula el tiempo disponible Td=11-10=1 y se actualiza la lista A.

Como el tiempo disponible en la estación es de 1 y no existe ninguna tarea que cumpla con ese valor se debe abrir otra estación, la cual es la numero 4, con el tiempo de ciclo completo de 21.

Tabla 116 Lista A 2.2.10

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
i	9	h	1
j	12	i	1

Tabla 117 Lista B 2.2.10

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0

Tabla 118 Lista C 2.2.10

Td=21

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0

Se asigna la tarea h a la estación 4, y se le calcula el tiempo disponible Td=21-8=13, y se actualiza la lista A.

Tabla 119 Lista A 2.2.11

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0
j	12	i	1

Tabla 120 Lista B 2.2.11

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0

Tabla 121 Lista C 2.2.11

Td=13

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0

Se asigna la tarea i a la estación numero 4, y se calcula el tiempo disponible de la estación Td=13-9=4, pero se debe abrir la estación numero 5 para poder asignar la tarea faltante, debido a que su tiempo de proceso es mayor al disponible en la estación.

Tabla 122 Estaciones iteración 2 SALB-2

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	c	10	10	5
	a	6	16	
2	b	11	11	4
	d	6	17	
3	e	7	7	1
	g	3	10	
	f	10	20	
4	h	8	8	4
	i	9	17	
5	j	12	12	9

La anterior iteración iniciando con un tiempo de ciclo de 21, arroja los siguientes resultados:

Tiempo de ciclo arrojado por el balanceo

C=20

Numero de estaciones

m=5

Eficiencia del balanceo

E=∑ Ti/ (m*c)

E=82/(5*20)

E=82%

Tiempo muerto total o demora del balance

D= (m*c)- ∑ Ti

D= (5*21)-82

D= 23

Esta iteración brinda 5 estaciones como las preestablecidas con un tiempo de ciclo de 20, mejorando la anterior respuesta, por lo que se guarda y queda como limite superior, aunque su eficiencia fue peor.

ITERACION 3

Para este caso se tomara un tiempo de ciclo de 19, para ver si se puede mejorar la anterior respuesta, teniendo en cuenta que el número de estaciones sigue siendo de 5.

Tabla 123 Lista A 2.3.1

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
b	11	a	1
c	10	-	0
d	6	b-c	2
e	7	d	1
h	8	e	1

g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 124 Lista B 2.3.1

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

Tabla 125 Lista C 2.3.1 Td=19

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

Como el Shift Ran dio el valor de 0,95, se escoge la tarea c para ser asignada a la estación número 1. Se calcula el tiempo disponible en la estación Td=19-10=9 y se actualiza la lista A.

Tabla 126 Lista A 2.3.2

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
b	11	a	1
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1

i	9	h-f-g	3
j	12	i	1

Tabla 127 Lista B 2.3.2

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0

Tabla 128 Lista C 2.3.2

Td=9

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0

Se asigna la tarea a en la estación 1, se calcula el tiempo disponible Td=9-6=3 y se actualiza la lista A.

Tabla 129 Lista A 2.3.3

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 130 Lista B 2.3.3

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0

Tabla 131 Lista C 2.3.3

Td=3

TAREA	Tj	PREDECE.	# PREDECE.
-	-	-	-

Como la lista C esta vacía, se abre la estación numero 2 con el tiempo de ciclo de 19 y se crea nuevamente la lista A.

Tabla 132 Lista A 2.3.4

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 133 Lista B 2.3.4

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0

Tabla 134 Lista C 2.3.4

Td=19

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0

La tarea b se asigna a la estación numero 2, se calcula el tiempo que se dispone en la estación Td=19-11= 8 y se actualiza la lista A.

Tabla 135 Lista A 2.3.5

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	O
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 136 Lista B 2.3.5

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	O

Tabla 137 Lista C 2.3.5

Td=8

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	O

Se asigna la tarea d a la estación número 2, se calcula el tiempo disponible Td=8-6=2 y se actualiza la lista A.

Tabla 138 lista A 2.3.6

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 139 Lista B 2.3.6

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0

Tabla 140 Lista C 2.3.6

Td=2

TAREA	Tj	PREDECE.	# PREDECE.
-	-	-	-

Como la lista C esta vacía, se abre la estación 3, con el tiempo completo de 19, y se crea la lista A.

Tabla 141 Lista A 2.3.7

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 142 Lista B 2.3.7

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0

Tabla 143 Lista C 2.3.7

Td=19

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0

Se asigna a la estación 3, la tarea e, se calcula el tiempo disponible en la estación Td=19-7=12 y se actualiza la lista A.

Tabla 144 Lista A 2.3.8

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0

i	9	h-f-g	3
j	12	I	1

Tabla 145 Lista B 2.3.8

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0

Tabla 146 Lista C 2.3.8

Td=12

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0

A través de Shift Ran que dio un valor de 0,155, se escoge la tarea h para asignarla a la estación 3, se calcula el tiempo disponible Td=12-8=4, y se actualiza lista A.

Tabla 147 Lista A 2.3.9

TAREA	Tj	PREDECE.	# PREDECE.
g	3	-	0
f	10	-	0
i	9	f-g	2
j	12	i	1

Tabla 148 Lista B 2.3.9

TAREA	Tj	PREDECE.	# PREDECE.
g	3	-	0
f	10	-	0

Tabla 149 Lista C 2.3.9

Td = 4

TAREA	Tj	PREDECE.	# PREDECE.
g	3	-	0

Se asigna la tarea g a la estación 3, se calcula el tiempo disponible Td=4-3=1 y se actualiza la lista A. Como ninguna tarea cumple con el tiempo disponible se abre otra estación.

Tabla 150 Lista A 2.3.10

TAREA	Tj	PREDECE.	# PREDECE.
f	10	-	0
i	9	f	1
j	12	i	1

Tabla 151 Lista B 2.3.10

TAREA	Tj	PREDECE.	# PREDECE.
f	10	-	0

Tabla 152 Lista C 2.3.10

Td=19

TAREA	Tj	PREDECE.	# PREDECE.
f	10	-	0

Se asigna la tarea f a la estación 4 y se calcula el tiempo disponible Td=19-10=9 y se actualiza la lista A.

Tabla 153 Lista A 2.3.11

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0
j	12	i	1

Tabla 154 Lista B 2.3.11

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0

Tabla 155 Lista C 2.3.11

Td=9

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0

Se asigna la tarea i a la estación 4, se calcula el tiempo disponible Td=9-9=0 y se actualiza la lista A.

Como únicamente falta por asignar la tarea j, y no queda tiempo disponible, se debe abrir una nueva estación para asignarla.

Tabla 156 Estaciones iteración 3 SALB -2

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	c	10	10	3
	a	6	16	
2	b	11	11	2
	d	6	17	
3	e	7	7	1
	h	8	15	
	g	3	18	
4	f	10	10	0
	i	9	19	
5	j	12	12	7

La anterior iteración iniciando con un tiempo de ciclo de 19, arroja los siguientes resultados:

Tiempo de ciclo del balanceo

C=19

Numero de estaciones

m=5

Eficiencia del balanceo

E=∑ Ti/ (m*c)

E=82/(5*19)

E=86.3%

Tiempo muerto total o demora del balance

D= (m*c)- ∑ Ti

D= (5*19)-82

D= 13

Esta iteración brinda 5 estaciones como las preestablecidas con un tiempo de ciclo de 19 mejorando la anterior respuesta, quedando como limite superior y

se desecha la anterior. Y también se puede observar que la eficiencia se mejora.

Ahora se hará otra iteración para mirar si se puede mejorar la anterior respuesta para ello se tomará un tiempo de ciclo de 18.

ITERACION 4

Tabla 157 Lista A 2.4.1

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
b	11	a	1
c	10	-	0
d	6	b-c	2
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 158 Lista B 2.4.1

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

Tabla 159 Lista C 2.4.1

Td=18

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
c	10	-	0

Se escoge aleatoriamente la tarea a asignar, a través de shift Ran, dando un valor de 0,587, que equivale a la tarea c, para asignarse a la estación 1.se calcula el tiempo disponible en la estación Td=18-10=8 y se actualiza la lista A.

Tabla 160 Lista A 2.4.2

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0
b	11	a	1
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 161 Lista B 2.4.2

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0

Tabla 162 Lista C 2.4.2

Td=8

TAREA	Tj	PREDECE.	# PREDECE.
a	6	-	0

Como solo esta la tarea a que cumple con el tiempo de la estación, se asigna a la estación 1. Se calcula el tiempo disponible Td=8-6=2 y se actualiza la lista A.

Tabla 163 Lista A 2.4.3

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 164 Lista B 2.4.3

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0

Tabla 165 Lista C 2.4.3

Td=2

TAREA	Tj	PREDECE.	# PREDECE.
-	-	-	-

La lista C esta vacía, por tanto se abre una nueva estación con el tiempo disponible completo de 18.

Tabla 166 Lista A 2.4.4

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0
d	6	b	1
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 167 Lista B 2.4.4

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0

Tabla 168 Lista C 2.4.4

Td=18

TAREA	Tj	PREDECE.	# PREDECE.
b	11	-	0

Se asigna la tarea b a la estación 2, el tiempo disponible es de Td=18-11=7 y se actualiza la lista A.

Tabla 169 Lista A 2.4.5

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	0
e	7	d	1
h	8	e	1
g	3	e	1
f	10	e	1
I	9	h-f-g	3
J	12	i	1

Tabla 170 Lista B 2.4.5

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	0

Tabla 171 Lista C 2.4.5

Td=7

TAREA	Tj	PREDECE.	# PREDECE.
d	6	-	0

Se asigna la tarea d a la estación 2, se calcula el tiempo disponible Td=7-6=1.

Observando el diagrama de precedencia, se nota que no existe tareas por asignar que cumplan con el tiempo disponible, por tanto se debe abrir una nueva estación, la numero 3.

Tabla 172 *Lista A 2.4.6*

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0
h	8	e	1
g	3	e	1
f	10	e	1
i	9	h-f-g	3
j	12	i	1

Tabla 173 Lista B 2.4.6

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0

Tabla 174 Lista C 2.4.6

Td=18

TAREA	Tj	PREDECE.	# PREDECE.
e	7	-	0

Se asigna la tarea e, a la estación 3 y se calcula el tiempo disponible Td=18-7=11 y se actualiza la lista A.

Tabla 175 Lista A 2.4.7

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0
i	9	h-f-g	3
j	12	i	1

Tabla 176 Lista B 2.4.7

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0

Tabla 177 Lista C 2.4.7

Td=11

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
f	10	-	0

A través del Shift Ran que arrojo un valor de 0,781 se escoge la tarea f, para asignarse a la estación 3. Se calcula el tiempo disponible Td=11-10=1.

Como el tiempo disponible en la estación es de 1 y no existe tareas que lo cumplan se debe abrir la cuarta estación, con un tiempo de ciclo de 18.

Tabla 178 Lista A 2.4.8

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0
i	9	h-g	2
j	12	i	1

Tabla 179 Lista B 2.4.8

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0

Tabla 180 Lista C 2.4.8

Td=18

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
g	3	-	0

Como Shift Ran dio 0,975, se escoge la tarea g para ser asignada a la estación 4. Se calcula el tiempo disponible Td=18-3=15, y se actualiza la lista A.

Tabla 181 Lista A 2.4.9

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0
i	9	h	1
j	12	i	1

Tabla 182 Lista B 2.4.9

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0

Tabla 183 Lista C 2.4.9

Td=15

TAREA	Tj	PREDECE.	# PREDECE.
h	8	-	0

Se asigna la tarea h a la estación 4, se calcula el tiempo disponible Td=15-8=7 y se actualiza la lista A.

Tabla 184 Lista A 2.4.10

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0
j	12	13	1

Tabla 185 Lista B 2.4.10

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0

Tabla 186 Lista C 2.4.10

Td=7

TAREA	Tj	PREDECE.	# PREDECE.
-	-	-	-

Como la lista C esta vacía se abre otra estación, la numero 5 con todo el tiempo disponible de 18 y se actualiza la lista A.

Tabla 187 Lista A 2.4.11

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0
j	12	i	1

Tabla 188 Lista B 2.4.11

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0

Tabla 189 Lista C 2.4.11

Td=18

TAREA	Tj	PREDECE.	# PREDECE.
i	9	-	0

Se asigna la tarea i a la estación 5, quedando con un tiempo de Td=18-9=9,pero al ver la única tarea que falta por asignar se nota que no cumple con el tiempo disponible por lo que se debe abrir la estación 6 para asignarse la tarea j, quedando un tiempo disponible en esa estación de 18-12=6.

Tabla 190 Estaciones iteración 4 SALB-2

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	c	10	10	2
	a	6	16	
2	b	11	11	1
	d	6	17	
3	e	7	7	1
	f	10	17	
4	g	3	3	7
	h	8	11	
5	i	9	9	9
6	j	12	12	6

La anterior iteración iniciando con un tiempo de ciclo de 18, arroja los siguientes resultados:

Tiempo de ciclo del balanceo

C=17

Numero de estaciones

m=6

Eficiencia del balanceo

E=∑ Ti/ (m*c)

E=82/(6*17)

E=80.39%

Tiempo muerto total o demora del balance

D= (m*c)- ∑ Ti

D= (6*18)-82

D= 26

Esta iteración arroja un menor valor de tiempo de ciclo, pero el número de estaciones fue mayor a la anterior iteración por lo que esta se desecha y queda la anterior respuesta como resultado final.

BRANCH AND BOUND Y WINQSB

Los problemas SALBP-1 y SALBP-2, se solucionaran a través del software de uso libre Winqsb 1.0

El Software es utilizado para la toma de decisiones y permite resolver muchos tipos de problemas en el campo de la investigación operativa. El programa contiene diferentes módulos a utilizar, dependiendo del tipo del problema, así:

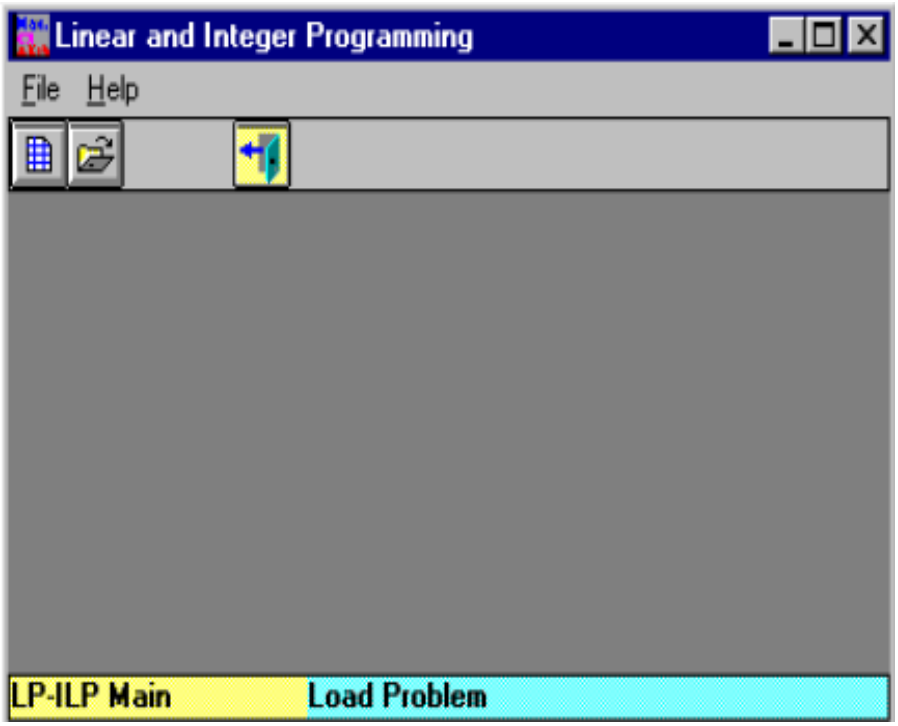
1. Linear Programming (LP) and integer linear programming (ILP)
2. Linear goal programming (GP) and integer linear programming (IGP).
3. Quadratic programming QP) and integer quadratic programming(IQP).
4. Network modeling (NET).
5. Nonlinear programming (NLP).
6. PERT/CPM

Este trabajo se centrará en el modulo 1 que permite resolver problemas de programación lineal entera utilizando el método del Branch and Bound.

11.1.3 RESOLUCION DEL SALBP-1 UTILIZANDO WINQSB

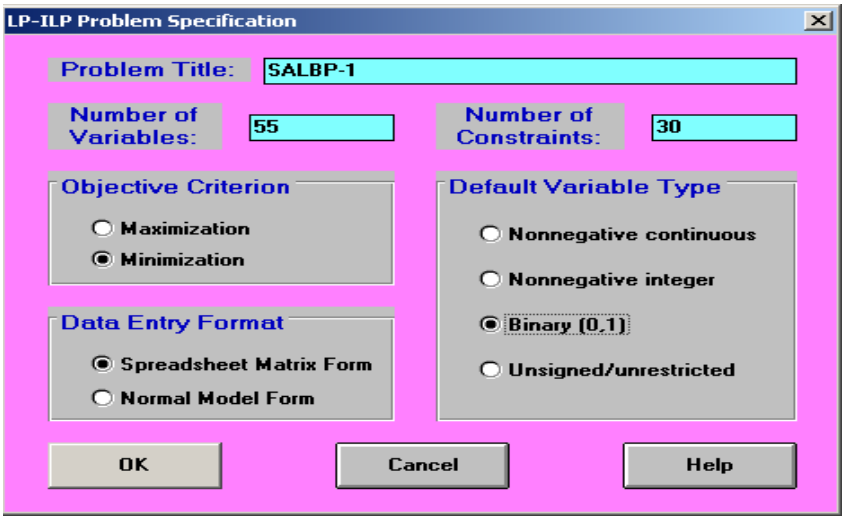
Al abrir el Software la primera ventana que aparece es la siguiente:

Figura 12. Pantalla inicial Winqsb



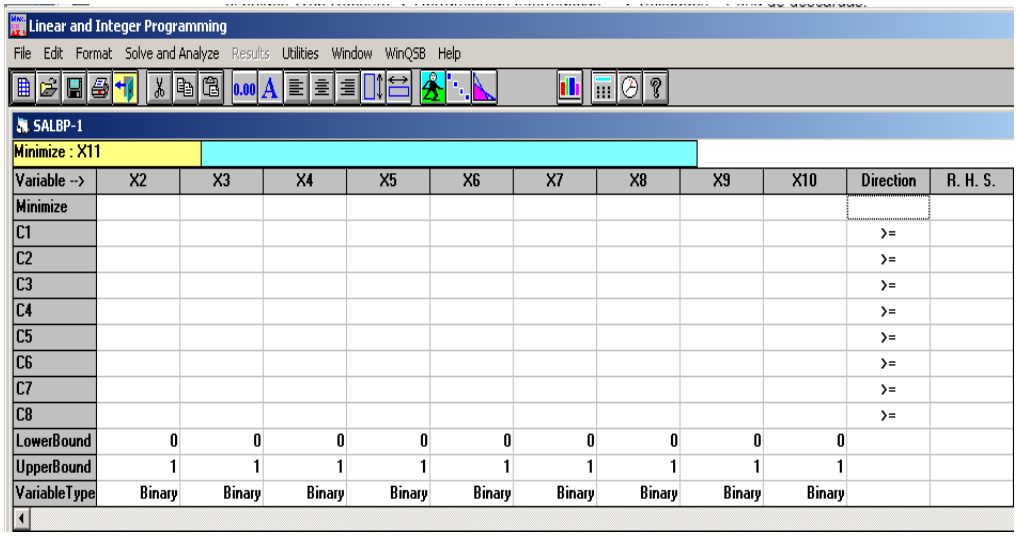
Por medio de esta ventana se puede crear un nuevo problema, a través del menú File o del botón  , surgiendo la siguiente ventana:

Figura 13. Ventana de especificación del problema



Aquí se coloca el nombre deseado, el número de variables, el número de restricciones, el objetivo y el tipo de variable. Después de llenar la información solicitada, se da click en OK y aparece la siguiente ventana:

Figura 14. Ventana para introducción de datos



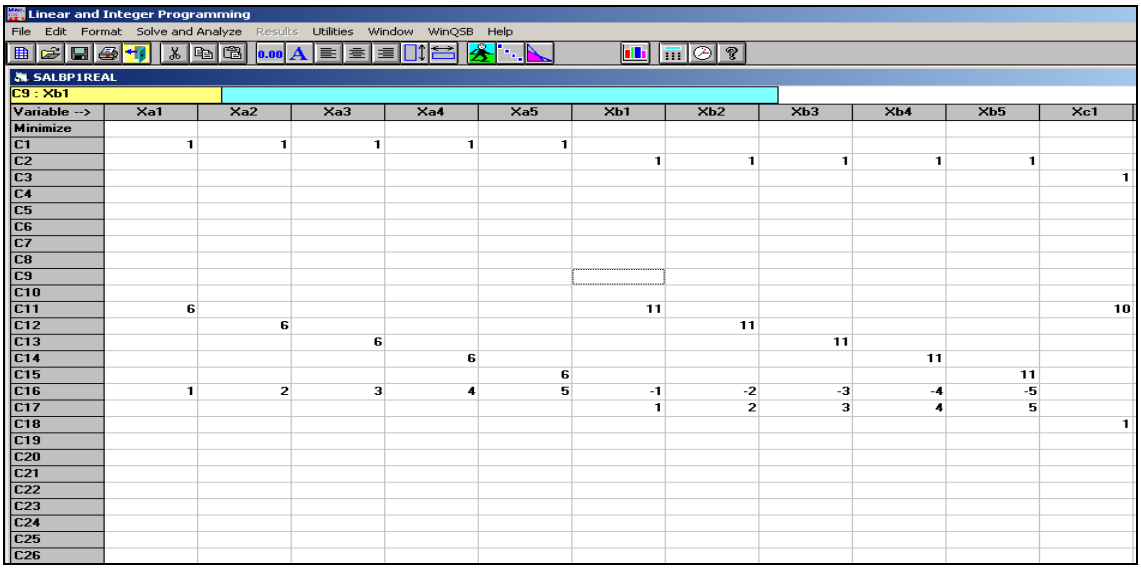
Variable -->	X2	X3	X4	X5	X6	X7	X8	X9	X10	Direction	R. H. S.
Minimize											
C1										>=	
C2										>=	
C3										>=	
C4										>=	
C5										>=	
C6										>=	
C7										>=	
C8										>=	
LowerBound	0	0	0	0	0	0	0	0	0		
UpperBound	1	1	1	1	1	1	1	1	1		
VariableType	Binary	Binary	Binary	Binary	Binary	Binary	Binary	Binary	Binary		

En esta ventana se introducen los datos del problema de la siguiente manera:

- En la primera fila de la matriz se introducen los coeficientes de la función objetivo.
- En el resto de las casillas se introducen los coeficientes de las restricciones.
- Para cambiar el sentido de las desigualdades o colocar una igualdad, se debe hacer doble clic sobre el símbolo deseado.

Después de tener todos los datos ingresados en la ventana, quedan de la siguiente manera (debido a que el problema es tan extenso solo se hace referencia a la parte inicial y final de la matriz).

Figura 15. Ventana parte inicial del problema SALBP-1



Linear and Integer Programming

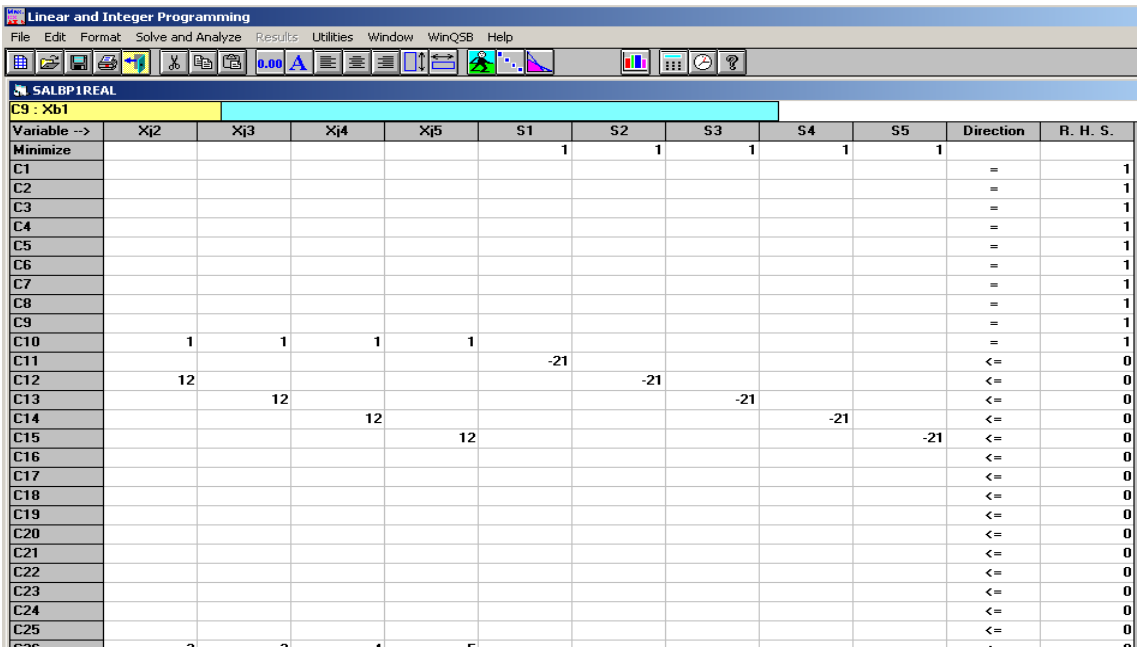
File Edit Format Solve and Analyze Results Utilities Window WinQSB Help

SALBP1REAL

C9 : Xb1

Variable -->	Xa1	Xa2	Xa3	Xa4	Xa5	Xb1	Xb2	Xb3	Xb4	Xb5	Xc1
Minimize											
C1	1	1	1	1	1						
C2						1	1	1	1	1	
C3											1
C4											
C5											
C6											
C7											
C8											
C9											
C10											
C11	6					11					10
C12		6					11				
C13			6					11			
C14				6					11		
C15					6					11	
C16	1	2	3	4	5	-1	-2	-3	-4	-5	
C17						1	2	3	4	5	
C18											1
C19											
C20											
C21											
C22											
C23											
C24											
C25											
C26											

Figura 16. Ventana parte final del problema SALBP-1



Linear and Integer Programming

File Edit Format Solve and Analyze Results Utilities Window WinQSB Help

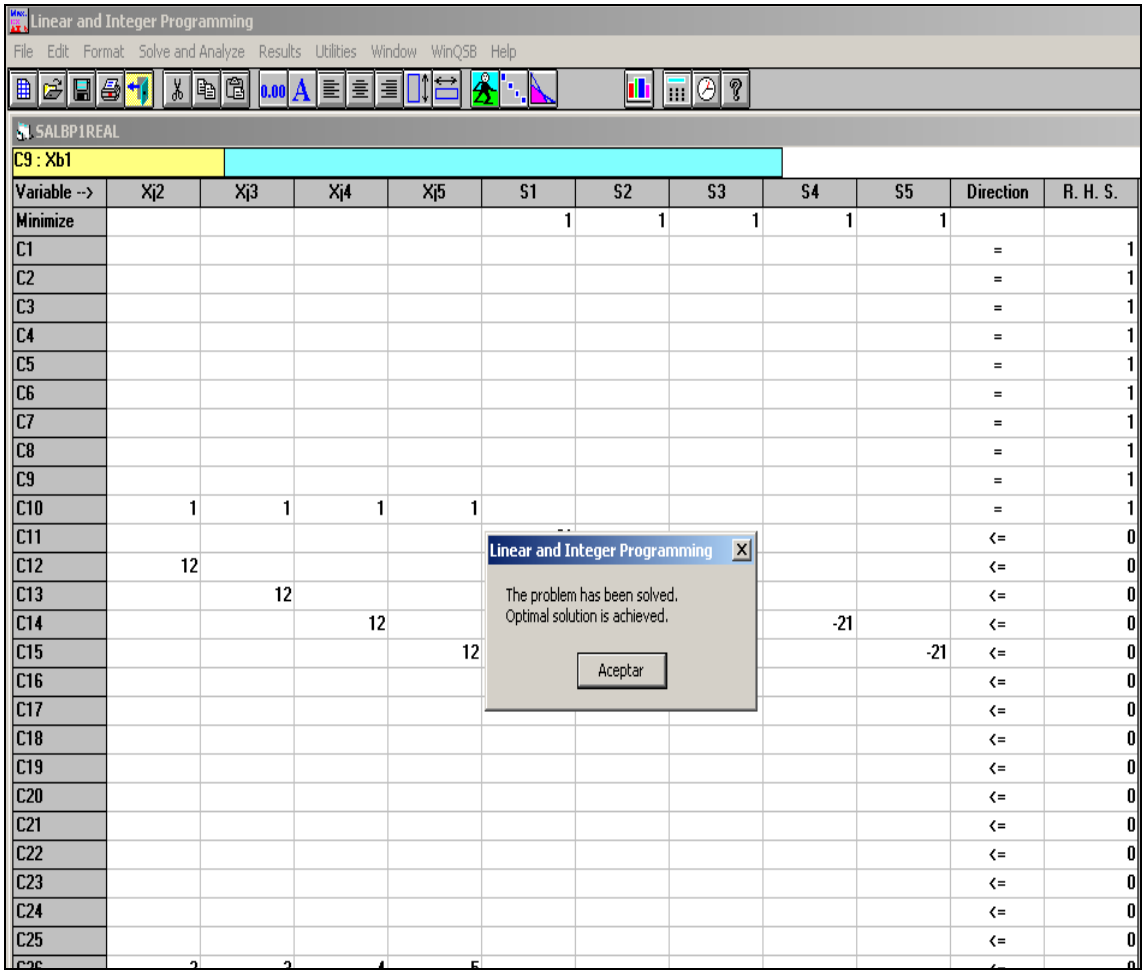
SALBP1REAL

C9 : Xb1

Variable -->	Xj2	Xj3	Xj4	Xj5	S1	S2	S3	S4	S5	Direction	R. H. S.
Minimize					1	1	1	1	1		
C1										=	1
C2										=	1
C3										=	1
C4										=	1
C5										=	1
C6										=	1
C7										=	1
C8										=	1
C9										=	1
C10	1	1	1	1						=	1
C11					-21					<=	0
C12	12					-21				<=	0
C13		12					-21			<=	0
C14			12					-21		<=	0
C15				12					-21	<=	0
C16										<=	0
C17										<=	0
C18										<=	0
C19										<=	0
C20										<=	0
C21										<=	0
C22										<=	0
C23										<=	0
C24										<=	0
C25										<=	0

Se da click en solve and analyze y nuevamente click en solve the problem, el programa realiza las iteraciones necesarias y muestra un cuadro de aviso donde se informa si el problema tiene solución o si es infactible, en este caso es factible y se muestra de la siguiente forma:

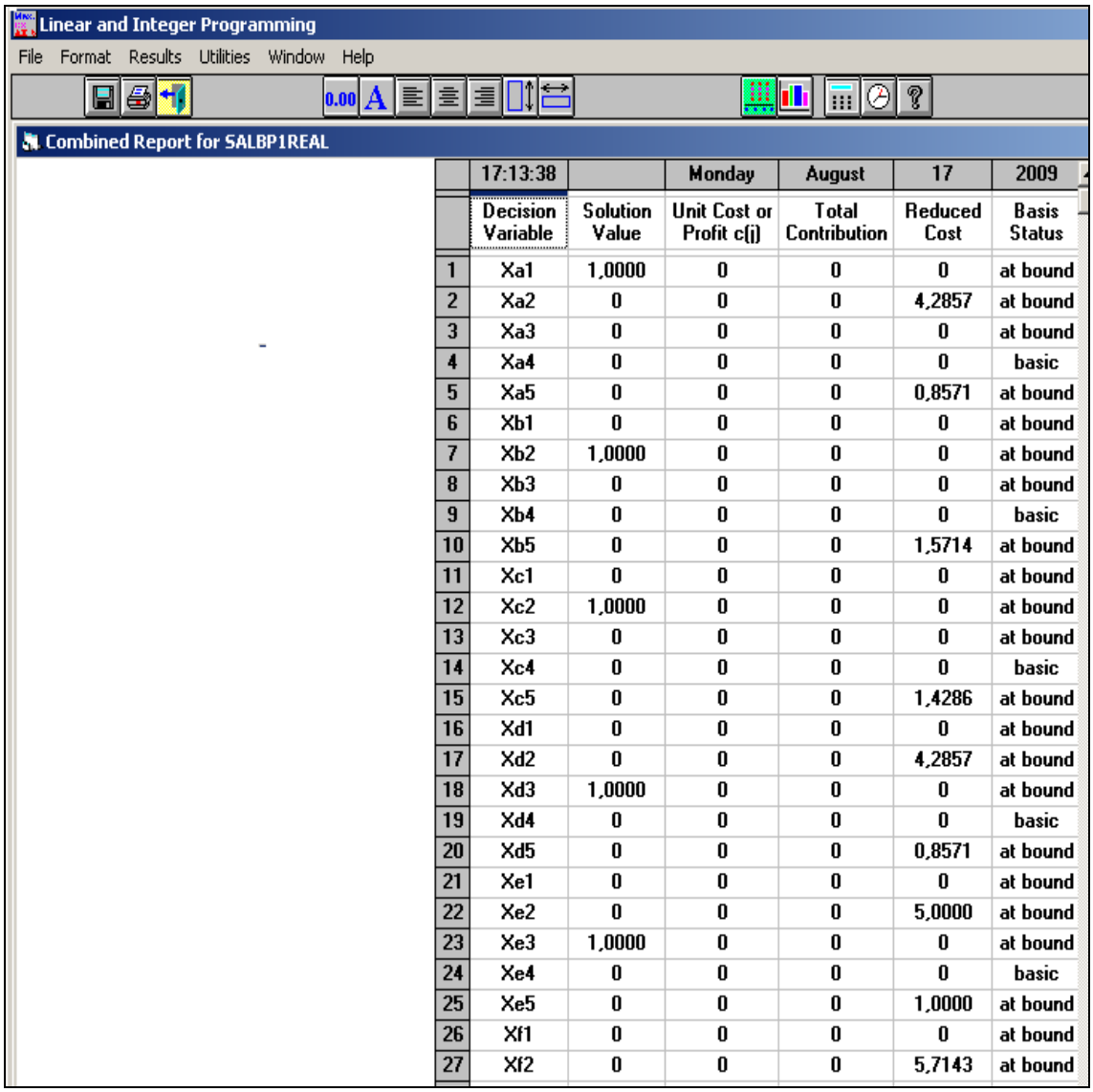
Figura 17. Ventana de confirmación Winqsb SALBP-1



Variable -->	Xj2	Xj3	Xj4	Xj5	S1	S2	S3	S4	S5	Direction	R. H. S.
Minimize					1	1	1	1	1		
C1										=	1
C2										=	1
C3										=	1
C4										=	1
C5										=	1
C6										=	1
C7										=	1
C8										=	1
C9										=	1
C10	1	1	1	1						=	1
C11										<=	0
C12	12									<=	0
C13		12								<=	0
C14			12					-21		<=	0
C15				12					-21	<=	0
C16										<=	0
C17										<=	0
C18										<=	0
C19										<=	0
C20										<=	0
C21										<=	0
C22										<=	0
C23										<=	0
C24										<=	0
C25										<=	0

Después de dar click en aceptar, se obtiene una tabla con los siguientes resultados.

Figura 18. Ventana parte inicial de resultados Winqsb SALBP-1



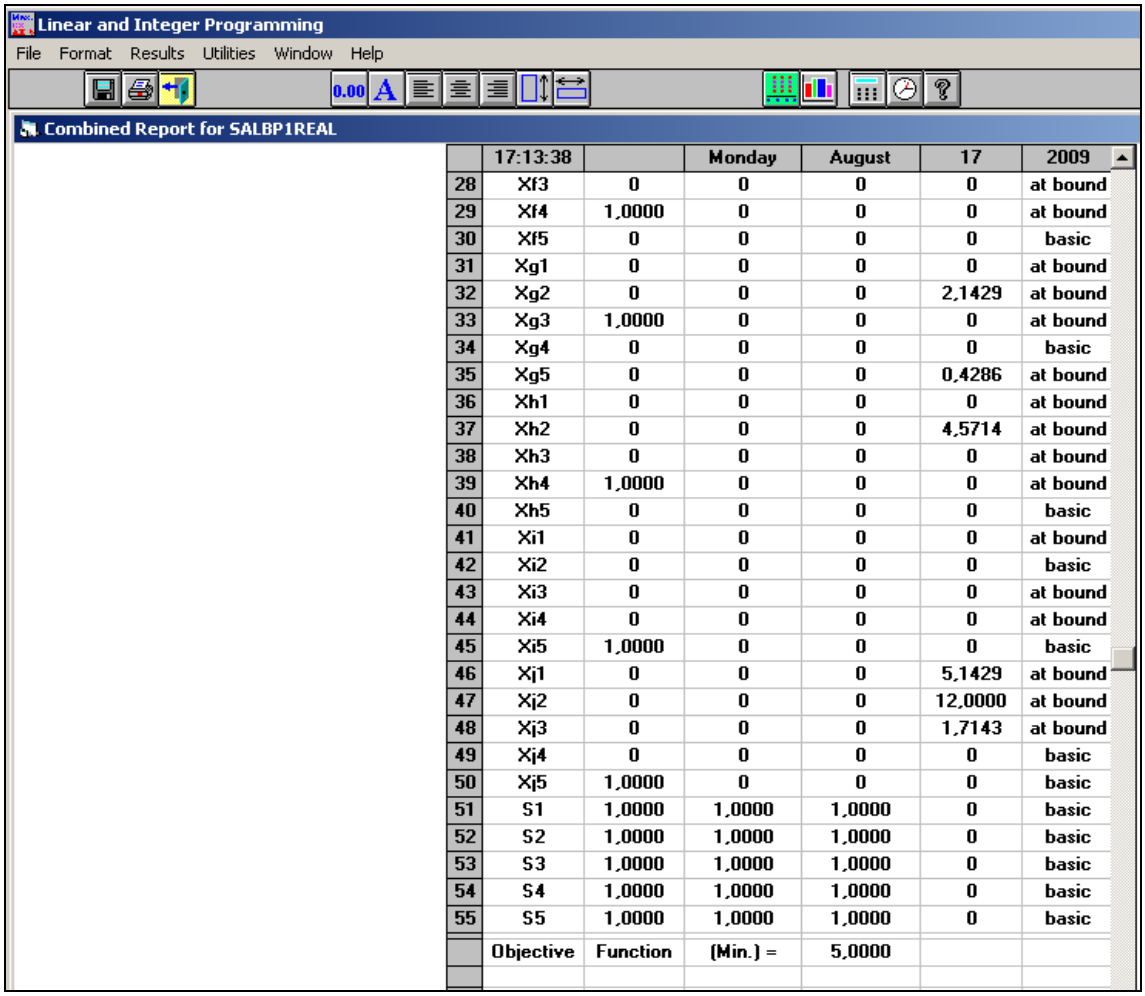
Linear and Integer Programming

File Format Results Utilities Window Help

Combined Report for SALBP1REAL

	17:13:38		Monday	August	17	2009
	Decision Variable	Solution Value	Unit Cost or Profit c[j]	Total Contribution	Reduced Cost	Basis Status
1	Xa1	1,0000	0	0	0	at bound
2	Xa2	0	0	0	4,2857	at bound
3	Xa3	0	0	0	0	at bound
4	Xa4	0	0	0	0	basic
5	Xa5	0	0	0	0,8571	at bound
6	Xb1	0	0	0	0	at bound
7	Xb2	1,0000	0	0	0	at bound
8	Xb3	0	0	0	0	at bound
9	Xb4	0	0	0	0	basic
10	Xb5	0	0	0	1,5714	at bound
11	Xc1	0	0	0	0	at bound
12	Xc2	1,0000	0	0	0	at bound
13	Xc3	0	0	0	0	at bound
14	Xc4	0	0	0	0	basic
15	Xc5	0	0	0	1,4286	at bound
16	Xd1	0	0	0	0	at bound
17	Xd2	0	0	0	4,2857	at bound
18	Xd3	1,0000	0	0	0	at bound
19	Xd4	0	0	0	0	basic
20	Xd5	0	0	0	0,8571	at bound
21	Xe1	0	0	0	0	at bound
22	Xe2	0	0	0	5,0000	at bound
23	Xe3	1,0000	0	0	0	at bound
24	Xe4	0	0	0	0	basic
25	Xe5	0	0	0	1,0000	at bound
26	Xf1	0	0	0	0	at bound
27	Xf2	0	0	0	5,7143	at bound

Figura 19. Ventana parte final de resultados Winqsb SALBP-1



Linear and Integer Programming

File Format Results Utilities Window Help

Combined Report for SALBP1REAL

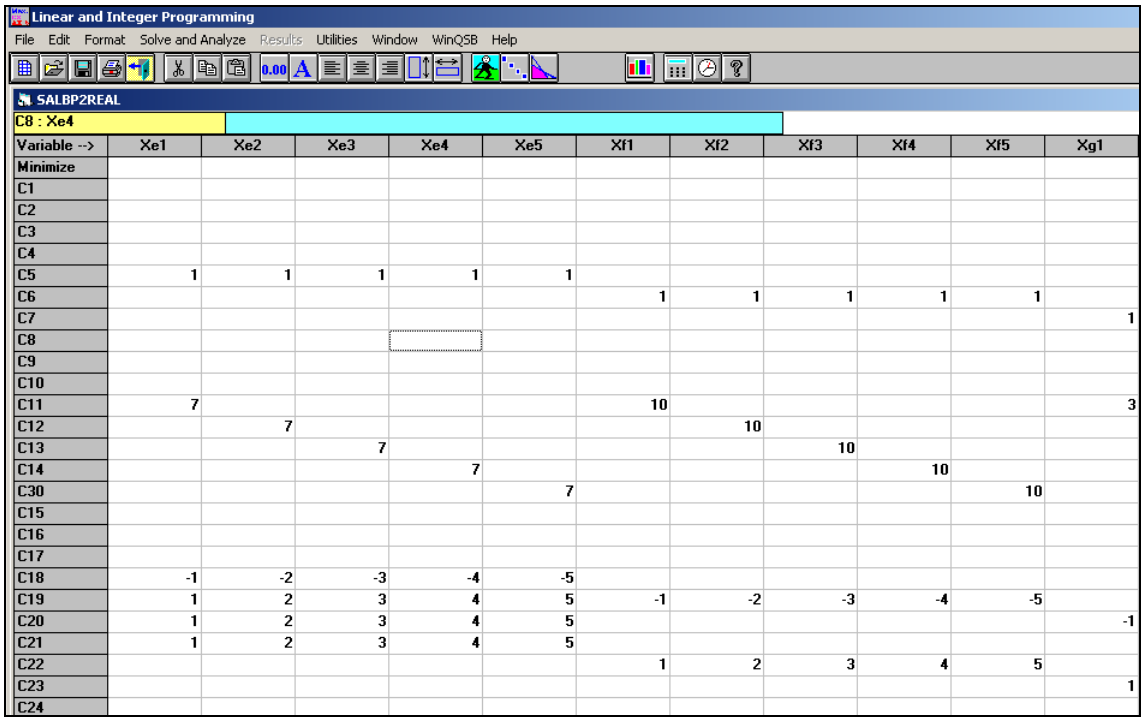
	17:13:38		Monday	August	17	2009
28	Xf3	0	0	0	0	at bound
29	Xf4	1,0000	0	0	0	at bound
30	Xf5	0	0	0	0	basic
31	Xg1	0	0	0	0	at bound
32	Xg2	0	0	0	2,1429	at bound
33	Xg3	1,0000	0	0	0	at bound
34	Xg4	0	0	0	0	basic
35	Xg5	0	0	0	0,4286	at bound
36	Xh1	0	0	0	0	at bound
37	Xh2	0	0	0	4,5714	at bound
38	Xh3	0	0	0	0	at bound
39	Xh4	1,0000	0	0	0	at bound
40	Xh5	0	0	0	0	basic
41	Xi1	0	0	0	0	at bound
42	Xi2	0	0	0	0	basic
43	Xi3	0	0	0	0	at bound
44	Xi4	0	0	0	0	at bound
45	Xi5	1,0000	0	0	0	basic
46	Xj1	0	0	0	5,1429	at bound
47	Xj2	0	0	0	12,0000	at bound
48	Xj3	0	0	0	1,7143	at bound
49	Xj4	0	0	0	0	basic
50	Xj5	1,0000	0	0	0	basic
51	S1	1,0000	1,0000	1,0000	0	basic
52	S2	1,0000	1,0000	1,0000	0	basic
53	S3	1,0000	1,0000	1,0000	0	basic
54	S4	1,0000	1,0000	1,0000	0	basic
55	S5	1,0000	1,0000	1,0000	0	basic
	Objective	Function	(Min.) =	5,0000		

Tabla 191 Resumen de estaciones SALBP-1 con Winqsb

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	a	6	6	15
2	b	11	11	0
	c	10	21	
3	d	6	6	1
	e	7	13	
	g	3	20	
4	f	10	10	3
	h	8	18	
5	i	9	9	0
	j	12	21	

11.1.4 RESOLUCION SALBP-2 UTILIZANDO WINQSB

Figura 20. Ventana parte inicial del problema SALBP-2



Linear and Integer Programming

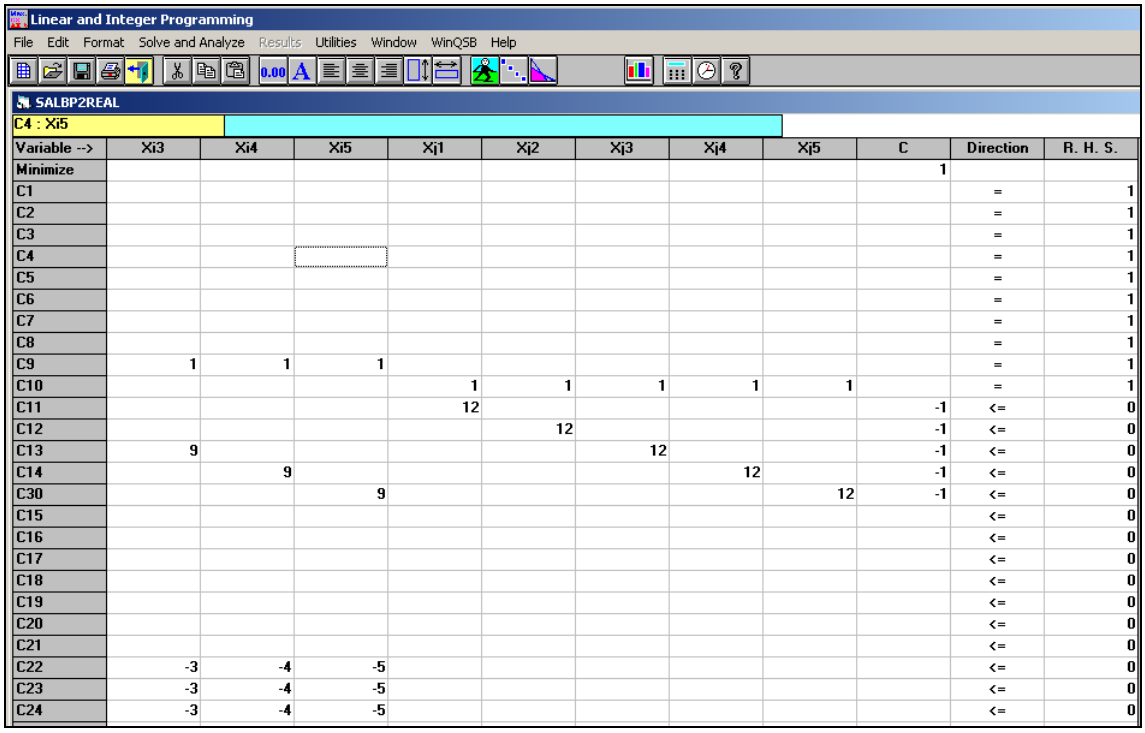
File Edit Format Solve and Analyze Results Utilities Window WinQSB Help

SALBP2REAL

C8 : Xe4

Variable -->	Xe1	Xe2	Xe3	Xe4	Xe5	Xf1	Xf2	Xf3	Xf4	Xf5	Xg1
Minimize											
C1											
C2											
C3											
C4											
C5	1	1	1	1	1						
C6						1	1	1	1	1	
C7											1
C8											
C9											
C10											
C11	7					10					3
C12		7					10				
C13			7					10			
C14				7					10		
C30					7					10	
C15											
C16											
C17											
C18	-1	-2	-3	-4	-5						
C19	1	2	3	4	5	-1	-2	-3	-4	-5	
C20	1	2	3	4	5						-1
C21	1	2	3	4	5						
C22						1	2	3	4	5	
C23											1
C24											

Figura 21. Ventana parte final del problema SALBP-2



Linear and Integer Programming

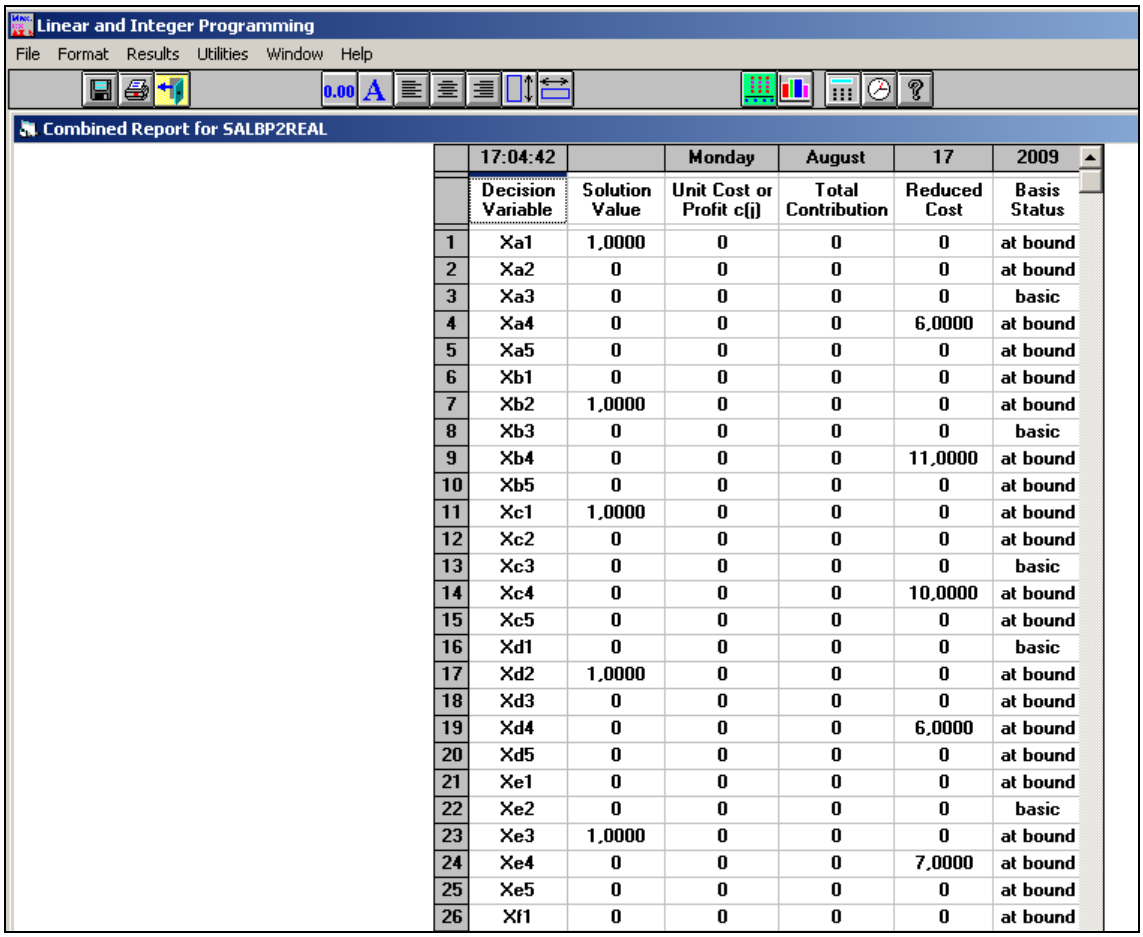
File Edit Format Solve and Analyze Results Utilities Window WinQSB Help

SALBP2REAL

C4 : Xi5

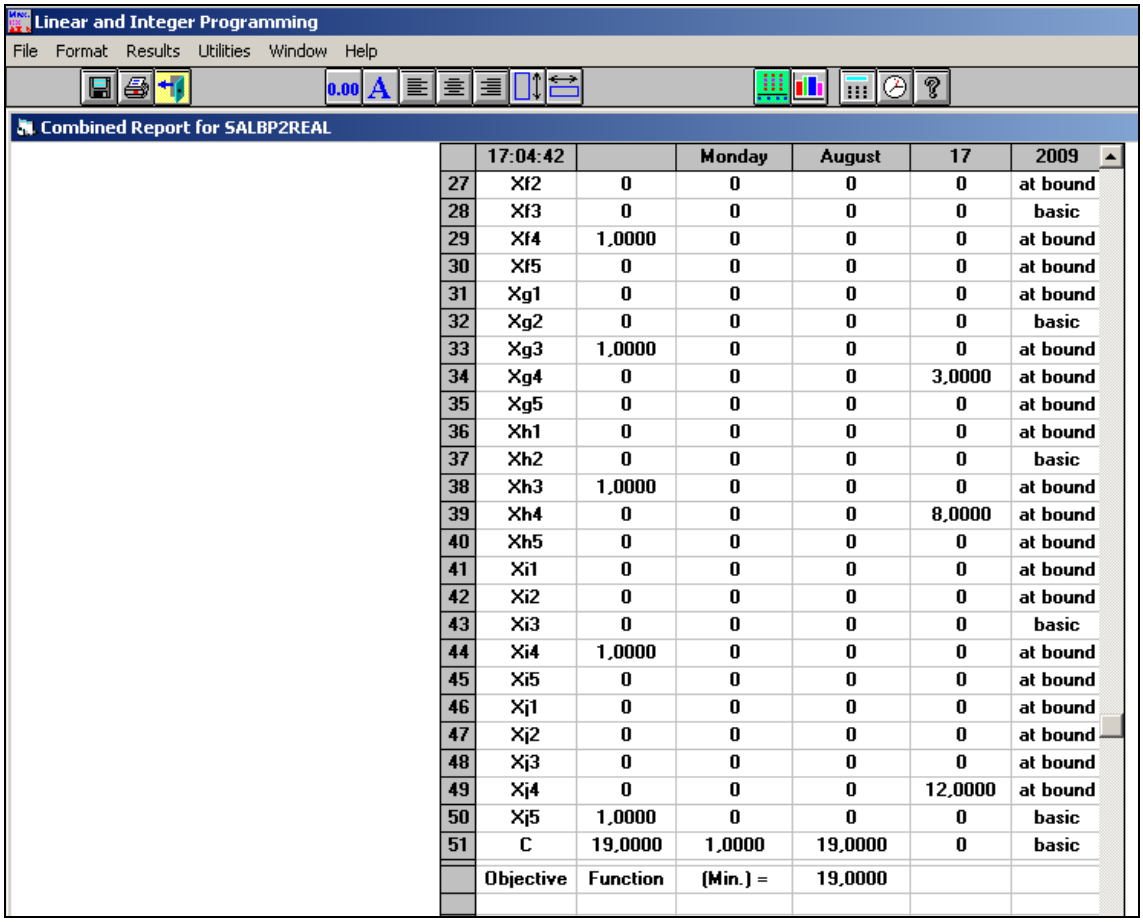
Variable -->	Xi3	Xi4	Xi5	Xj1	Xj2	Xj3	Xj4	Xj5	C	Direction	R. H. S.
Minimize									1		
C1										=	1
C2										=	1
C3										=	1
C4										=	1
C5										=	1
C6										=	1
C7										=	1
C8										=	1
C9	1	1	1							=	1
C10				1	1	1	1	1		=	1
C11				12					-1	<=	0
C12					12				-1	<=	0
C13	9					12			-1	<=	0
C14		9					12		-1	<=	0
C30			9					12	-1	<=	0
C15										<=	0
C16										<=	0
C17										<=	0
C18										<=	0
C19										<=	0
C20										<=	0
C21										<=	0
C22	-3	-4	-5							<=	0
C23	-3	-4	-5							<=	0
C24	-3	-4	-5							<=	0

Figura 22. Ventana inicial de resultados Winqsb SALBP-2



	17:04:42		Monday	August	17	2009
	Decision Variable	Solution Value	Unit Cost or Profit c(j)	Total Contribution	Reduced Cost	Basis Status
1	Xa1	1,0000	0	0	0	at bound
2	Xa2	0	0	0	0	at bound
3	Xa3	0	0	0	0	basic
4	Xa4	0	0	0	6,0000	at bound
5	Xa5	0	0	0	0	at bound
6	Xb1	0	0	0	0	at bound
7	Xb2	1,0000	0	0	0	at bound
8	Xb3	0	0	0	0	basic
9	Xb4	0	0	0	11,0000	at bound
10	Xb5	0	0	0	0	at bound
11	Xc1	1,0000	0	0	0	at bound
12	Xc2	0	0	0	0	at bound
13	Xc3	0	0	0	0	basic
14	Xc4	0	0	0	10,0000	at bound
15	Xc5	0	0	0	0	at bound
16	Xd1	0	0	0	0	basic
17	Xd2	1,0000	0	0	0	at bound
18	Xd3	0	0	0	0	at bound
19	Xd4	0	0	0	6,0000	at bound
20	Xd5	0	0	0	0	at bound
21	Xe1	0	0	0	0	at bound
22	Xe2	0	0	0	0	basic
23	Xe3	1,0000	0	0	0	at bound
24	Xe4	0	0	0	7,0000	at bound
25	Xe5	0	0	0	0	at bound
26	Xf1	0	0	0	0	at bound

Figura 23. Ventana final de resultados Winqsb SALBP-2



	17:04:42		Monday	August	17	2009
27	Xf2	0	0	0	0	at bound
28	Xf3	0	0	0	0	basic
29	Xf4	1,0000	0	0	0	at bound
30	Xf5	0	0	0	0	at bound
31	Xg1	0	0	0	0	at bound
32	Xg2	0	0	0	0	basic
33	Xg3	1,0000	0	0	0	at bound
34	Xg4	0	0	0	3,0000	at bound
35	Xg5	0	0	0	0	at bound
36	Xh1	0	0	0	0	at bound
37	Xh2	0	0	0	0	basic
38	Xh3	1,0000	0	0	0	at bound
39	Xh4	0	0	0	8,0000	at bound
40	Xh5	0	0	0	0	at bound
41	Xi1	0	0	0	0	at bound
42	Xi2	0	0	0	0	at bound
43	Xi3	0	0	0	0	basic
44	Xi4	1,0000	0	0	0	at bound
45	Xi5	0	0	0	0	at bound
46	Xj1	0	0	0	0	at bound
47	Xj2	0	0	0	0	at bound
48	Xj3	0	0	0	0	at bound
49	Xj4	0	0	0	12,0000	at bound
50	Xj5	1,0000	0	0	0	basic
51	C	19,0000	1,0000	19,0000	0	basic
	Objective	Function	(Min.) =	19,0000		

Tabla 192 Resumen de estaciones SALBP-2 con Winqsb

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	a	6	6	3
	c	10	16	
2	b	11	11	2
	d	6	17	
3	e	7	7	1
	g	3	10	
	h	8	18	
4	f	10	10	0
	i	9	19	
5	j	12	12	7

11.2 COMPARACION Y ANALISIS DE RESULTADOS

11.2.1 Comparación de resultados obtenidos con COMSOAL Y B&B con Winqsb para el SALBP-1

Tabla 193 Comparación de resultados SALBP-1

Nº estación	COMSOAL		Branch and Bound(Winqsb)	
	$T_{operación}$	T_{ocioso}	$T_{operación}$	T_{ocioso}
1	17	5	6	15
2	16	4	21	0
3	18	3	20	1
4	19	2	18	3
5	12	9	21	0
Tiempo muerto total	23		19	
Eficiencia real de la línea	86.3%		78.09%	

Figura 24. Grafica comparativa de los tiempos de operación por estación para el SALBP-1

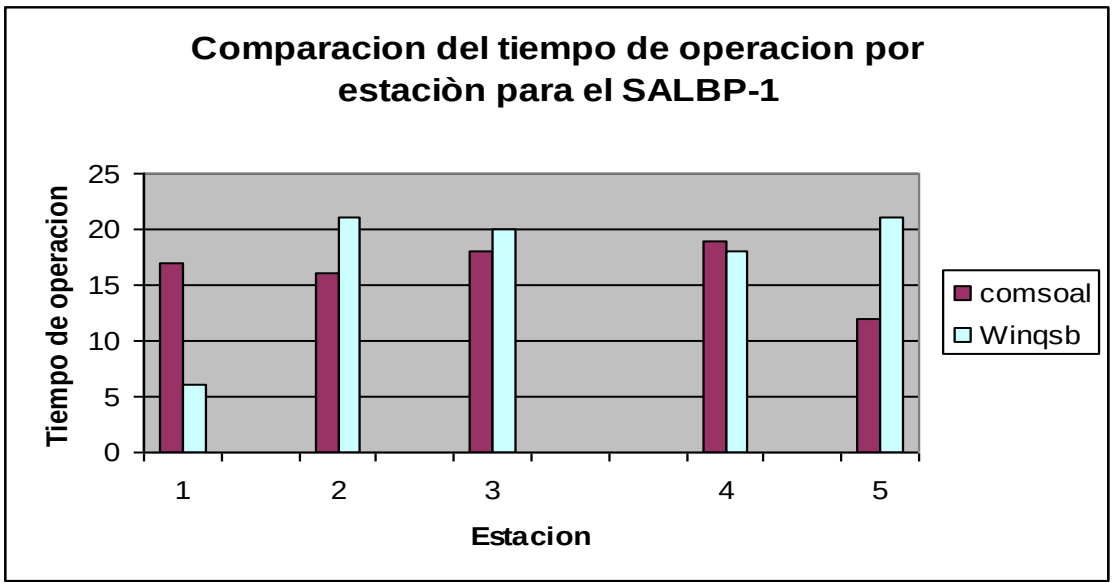
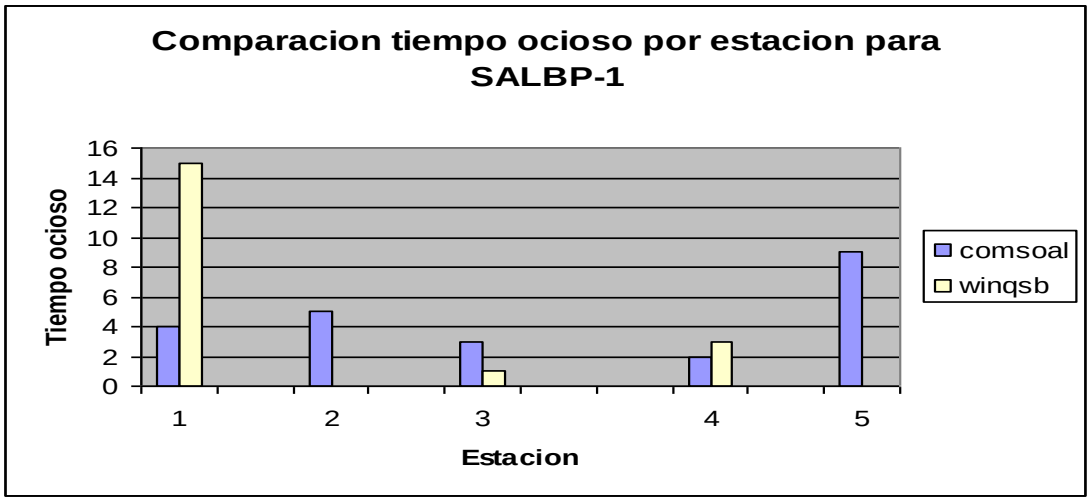


Figura 25. Grafica comparativa del tiempo ocioso por estación para SALBP-1



Análisis SALBP-1

Como muestran los resultados obtenidos, el tiempo ocioso total es menor en el Branch and Bound, sin embargo se puede decir que el algoritmo Comsoal ofrece una mejor solución debido a que arrojo un menor tiempo de ciclo y su eficiencia es mejor, adicionalmente la carga de trabajo esta mejor distribuída.

11.2.2 Comparación de resultados obtenidos con COMSOAL Y B&B con Winqsb para el SALBP-2

Tabla 194 Comparación de resultados SALBP-2

Nº estación	COMSOAL		Branch and Bound(Winqsb)	
	$T_{operación}$	T_{ocioso}	$T_{operación}$	T_{ocioso}
1	16	3	16	3
2	17	2	17	2
3	18	1	18	1
4	19	0	19	0
5	12	7	12	7
Tiempo muerto total	13		13	
Eficiencia real de la línea	86.3%		86.3%	

Figura 26. Grafica comparativa del tiempo de operación por estación para SALBP-2

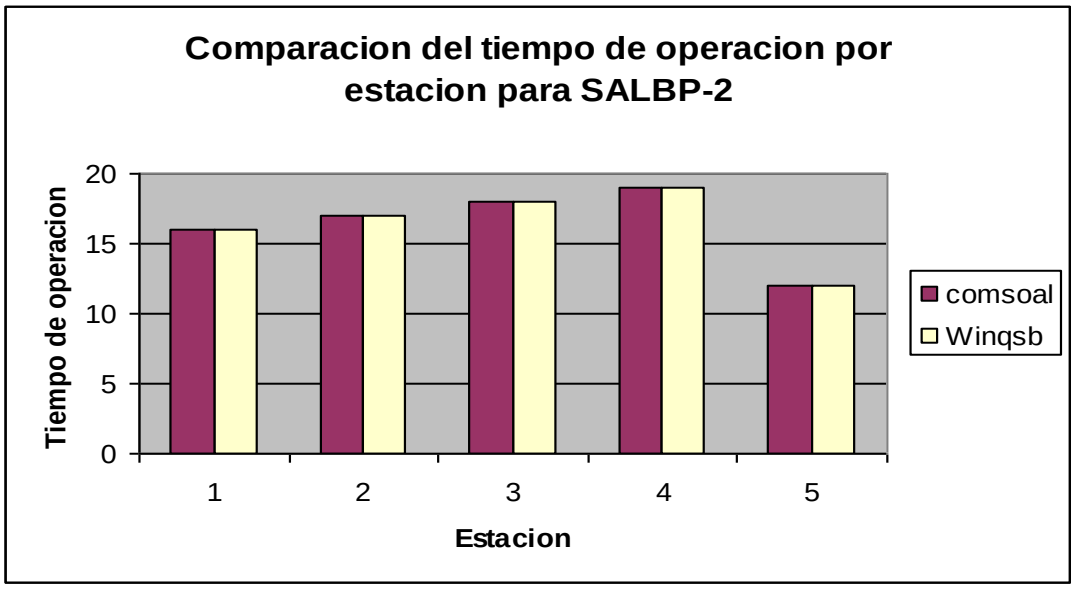
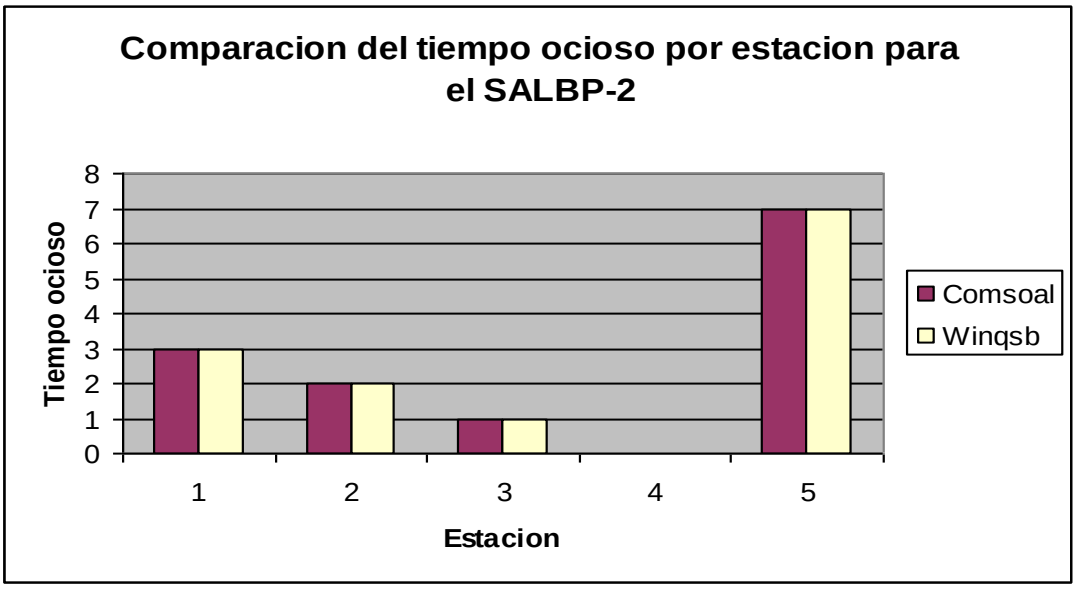


Figura 27. Grafica comparativa del tiempo ocioso por estación para SALBP-2



Análisis SALBP-2

Para este caso especifico, los resultados obtenidos fueron similares.

12. CONCLUSIONES

Esta investigación tuvo como objetivo primordial establecer las herramientas mas usadas en el campo de la investigación de operaciones para el balanceo de línea simple basada en una revisión bibliográfica acerca del tema, por supuesto el balanceo es un tema de investigación muy amplio y todavía en desarrollo.

Para este estudio se abordaron dos técnicas muy conocidas como son el COMSOAL y el BRANCH AND BOUND a través de un pequeño problema específico con el fin de solucionar dos problemas que se presentan en balanceo, minimizar el número de estaciones y minimizar el tiempo de ciclo. El documento deja claro que el algoritmo COMSOAL brinda muy buenas soluciones, es más práctico y más recomendable ya que existen software especializados de fácil manejo y un óptimo desempeño en comparación al programa Winqsb, el cual esta diseñado para otro tipo de aplicaciones. En el caso de aquellos problemas que tienen muchas variables el Winqsb se vuelve poco práctico al momento de ingresar los datos y su resolución es lenta debido a la gran cantidad de iteraciones que debe realizar.

El desarrollo de esta investigación permite conocer la importancia de implementar la teoría aprendida a través de un trabajo exigente que enseña con una metodología clara el proceso de balanceo y como resultado apoya el proceso de formación profesional.

13. BIBLIOGRAFIA

CAPACHO L, MORENO R, Generación de secuencias de montaje y equilibrado de líneas, Universidad Politécnica de Catalunya, Abril 2004.

RESTREPO J, MEDINA P, CRUZ E, Balanceo de un modulo de confección utilizando el algoritmo de Helgeson and Birnie: Un caso de estudio, Universidad Tecnológica de Pereira, Diciembre 2006

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

HILLIER, LIEBERMAN. Investigacion de operaciones 7 edicion pag.604

INTERNET

http://es.wikipedia.org/wiki/Heur%C3%ADstica

http://www.freequality.org/sites/www_freequality_org/documents/Training/Classes%2520Spring%25202003/PowerPoint%2520Slides/COMSOAL.ppt+comsoal+e+IBM&hl=es&ct=clnk&cd=1&gl=co

http://racero.us.es/Asignaturas/Secuenciacion/EQUILIBRADO.PPT (septiembre 2008)

www.velocidadmaxima.com/forum/showthread.php?t=1187,marzo 2009

https://upcommons.upc.edu/pfc/bitstream/2099.1/2587/1/31026-1.pdf,septiembre 2008.

http://descartes.cnice.mec.es/materiales_didacticos/prog_lineal_lbc/definicion_pl.htm,septiembre 2008.

http://www.investigacion-operaciones.com/Libro/Programacion%20Entera.pdf

https://upcommons.upc.edu/pfc/bitstream/2099.1/2587/1/31026-1.pdf,agosto 2008

http://www.eumed.net/libros/2006c/216/1j.htm,agosto 2008

[1] https://upcommons.upc.edu/pfc/bitstream/2099.1/2587/1/31026-1.pdf, agosto 2008

[1] http://www.utp.edu.co/php/revistas/ScientiaEtTechnica/docsFTP/18521325-330.pdf

https://upcommons.upc.edu/pfc/bitstream/2099.1/2587/1/31026-1.pdf,noviembre 2008

http://www.icaen.uiowa.edu/~dbricker/Stacks_pdf2/ALB_COMSOAL.pdf.noviembre 2008

14. BIBLIOGRAFIA COMPLEMENTARIA

Arcus, A.L. COMSOAL: A computer method of sequencing operations for assembly lines, International Journal of Production Research 4, 259-277. 1966.

Bard, J. Assembly line balancing with parallel workstations and dead times. International Journal of Production Research. Vol. 37, No. 4, 721-736. 1999.

Bartholdi, J.J. Balancing two-sided assembly lines: A case study, International Journal of Production Research 31, 2447-2461. 1993.

Bautista, J. y Pereira, J. Algoritmos de hormigas para un problema de equilibrado de líneas. V Congreso de Ingeniería de Organización, Valladolid-Burgos. 2003.

Baybars, I. A survey of exact algorithms for the simple assembly line balancing problem" Management Science, 32, 909-932, 1986.

Becker, C. y Scholl, A. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. Technical Report 20/2003; Friedrich-Schiller-Universität Jena (Invited review for the special issue: "Balancing of automated assembly and transfer lines" of the European Journal of Operational Research), 2003.

Scholl, A. y Voss, S. Simple assembly line balancing – Heuristic approaches. Journal of Heuristics, 2, 217-244, 1996.

Scholl, A. y Klein, R. SALOME: A bidirectional branch and bound procedure for assembly line balancing. INFORMS Journal on Computing, 9, 319-334, 1997.

Scholl, A. Balancing and sequencing of Assembly lines. Physica-Verlag,1999.

Scholl, A. y Klein, R. ULINO: Optimally balancing U-shaped JIT assembly lines. International Journal of Production Research. Vol. 37, No. 4, 721-736. 1999.

APLICACIÓN DE ALGUNOS MÉTODOS EXACTOS Y HEURISTICOS PARA RESOLVER EL PROBLEMA DE BALANCEO DE LINEA SIMPLE

Autores
JAVIER ANDRES PÓVEDA ZAPATA
10.004.424
CARLOS HERNANDO FLOREZ HURTADO
9.861.610

RESUMEN

Este documento investigativo pretende dar solución a un ejemplo particular de balanceo de línea simple, el cual se estudiara a través de dos tipos de problemas denominados SALBP-1 y SALBP-2, por medio de un método heurístico conocido como COMSOAL y un método exacto como lo es el BRANCH & BOUND.

OBJETIVOS

OBJETIVO GENERAL

Solucionar el problema de balanceo de línea simple tipo SALBP-1 y SALBP-2 con el método COMSOAL y BRANCH & BOUND (con Winqsb Versión 1.0).

OBJETIVOS ESPECÍFICOS

- Definir el problema de balanceo de línea.
- Clasificar el problema.
- Plantear el modelo matemático del problema.
- Solucionar con COMSOAL.
- Solucionar con BRANCH AND BOUND (Utilizando Winqsb Version 1.00).

DEFINICIÓN DEL PROBLEMA

- Como es sabida una línea bien equilibrada trae no solo reducción en los costos de operación, sino también un mejor nivel de calidad y por lo tanto un aumento en la satisfacción de los clientes, por otro lado la competitividad se mejora notablemente, viéndose reflejado este incremento en las ventas y en la imagen de las empresas.
- Se debe reconocer los dos tipos de problema de balanceo de línea, los cuales son: el simple ó SALBP (simple assembly line balancing problem) y el general ó GALBP (general assembly line balancing problem).
- En este tipo de problema se consideran líneas de producción simples en las que las estaciones son colocadas en serie y solo existe un solo tipo de producto con tiempos conocidos, además las estaciones pueden realizar cualquier tipo de tarea y las tareas pueden ser realizadas en cualquier estación, los tiempos en el problema simple son determinìsticos y desde luego conocidos con anticipación.

CLASIFICACIÓN DE LOS SALBPS

- SALBP -1: El problema reside en hallar una solución factible que permita minimizar el número de estaciones, dado un tiempo de ciclo. Este tipo se da cuando se va a hacer un nuevo montaje y la demanda puede ser estimada.
- SALBP-2: Busca minimizar el tiempo de ciclo dado un número de estaciones fijas, aquí se supone que la línea existe.
- SALBP-E: este problema busca maximizar la eficiencia de la línea, es decir busca minimizar el producto de m (numero de estaciones) por c (tiempo de ciclo).
- SALBP-F: se busca una solución factible dado el tiempo de ciclo y el numero de estaciones.

PLANTEAMIENTO DEL PROBLEMA

Modelación matemática SALBP-1

$x_{ij} = 1$ *si operación i se hace en estación j*

$y_j = 1$ *si existe estación j*

$$Min\, Z = \sum_{j=1}^{M_{max}} y_j$$

$$s.a$$

$$(1)\quad \sum_{i=1}^{N} t_i \cdot x_{ij} \le C \cdot y_j \qquad j = 1,\ldots, M_{\max}$$

$$(2)\quad \sum_{j=1}^{M_{max}} x_{ij} = 1 \qquad i = 1,\ldots, N$$

$$(3)\quad \sum_{j=1}^{M_{max}} j \cdot x_{kj} \le \sum_{j=1}^{M_{max}} j \cdot x_{ij} \qquad \forall k \prec i$$

$$(4)\quad y_{j+1} \le y_j \qquad j = 1,\ldots, M_{max} - 1$$

$$x_{ij} = \{0,1\} \quad \forall (i,j); \qquad y_j = \{0,1\} \quad \forall (j)$$

PLANTEAMIENTO DEL PROBLEMA

Modelación matemática SALBP-2

$x_{ij} = 1$ *si operación i se hace en estación j*

C *tiempo de ciclo*

$$Min\, C$$

$$s.a$$

$$(1)\sum_{i=1}^{N} t_i \cdot x_{ij} \leq C \quad j = 1,\ldots, M$$

$$(2)\sum_{j=1}^{M} x_{ij} = 1 \quad i = 1,\ldots, N$$

$$(3)\sum_{j=1}^{M} j \cdot x_{kj} \leq \sum_{j=1}^{M} j \cdot x_{ij} \quad \forall k \prec i$$

$$x_{ij} = \{0,1\} \quad \forall (i, j); \quad C \in \Re^{+}$$

COMSOAL

COMSOAL

- Computer Method for sequencing operation for assembly lines.
- En español sus siglas significan, Método Computarizado para la secuencia de operaciones en la línea de ensamble.

COMSOAL

PASOS

- PASO 1: Para cada tarea se debe identificar las tareas que le siguen en orden de precedencia.
- PASO 2: Crear una LISTA A, que consiste en colocarle a cada tarea de la línea de ensamble, el numero de tareas que le preceden.
- PASO 3: De la lista A, se crea una lista B, con las tareas que tienen cero predecesores, si no hay tareas por asignar entonces se debe parar.

COMSOAL

- PASO 4: De la lista B, se crea la lista C, compuesta por aquellas tareas cuyos tiempos de proceso no sea mayor al tiempo de ciclo disponible en la estación. Si esta lista esta vacía se debe abrir una nueva estación, la cual tendrá nuevamente todo el tiempo de ciclo disponible y se repite el paso 4.

- PASO 5: De forma aleatoria se escoge de la lista C la tarea a asignarse a la estación.

- PASO 6: Se debe actualizar el tiempo disponible en la estación y la lista B, con el fin de mirar el tiempo consumido y los predecesores completados hasta el momento. Si la lista B esta vacía se debe actualizar la lista A, y se vuelve al paso 3, de lo contrario se debe regresar al paso 4

COMSOAL

VENTAJAS

- Permite examinar un número grande de secuencias con un simple registro encontrando soluciones factibles y en poco tiempo.
- Es una técnica fácil de programar.
- El método solo tiene en cuenta aquellas tareas que cumplen con todas las restricciones en cada paso.
- Una secuencia es descartada cuando excede el límite superior.
- Una secuencia es guardada cuando se mejora el límite superior anterior.
- Las secuencias son generadas al escoger aleatoriamente una tarea y construye subsecuentes tareas.
- Nuevas estaciones son abiertas cuando se necesitan.

BRANCH AND BOUND

- Técnica conocida por sus aplicaciones a los problemas de programación entera.
- la técnica se fundamenta en la idea de dividir y vencerás.
- Para hacer el problema inicial o principal este se divide en subproblemas cada vez más pequeños.

BRANCH AND BOUND

Pasos

Ramificación:

- Entre los subproblemas (no sondeados) se elige el de creación mas reciente, se ramifica el nodo en dos subproblemas fijando la variable de ramificación.

Acotamiento:

- Para cada nuevo subproblema se obtiene su cota aplicando por ejemplo el método simplex u otro método pertinente a su soltura de PL y redondeando hacia abajo el valor de Z en la solución optima.

Sondeo:

- Para cada nuevo subproblema se aplican las tres pruebas de sondeo y se descartan aquellos problemas que quedan sondeados o eliminados por cualquiera de las tres pruebas.

BRANCH AND BOUND

Existen tres formas de sondear:

1 Prueba: su cota <= Z *

2 Prueba: su soltura de PL no tiene soluciones factibles.

3 Prueba : la solución optima para su soltura de PL es entera (si esta solución es mejor que la de apoyo , se convierte en la nueva solución de apoyo y se aplica de nuevo la prueba 1 a todos los subproblemas no sondeados, con la nueva Z * mejor).

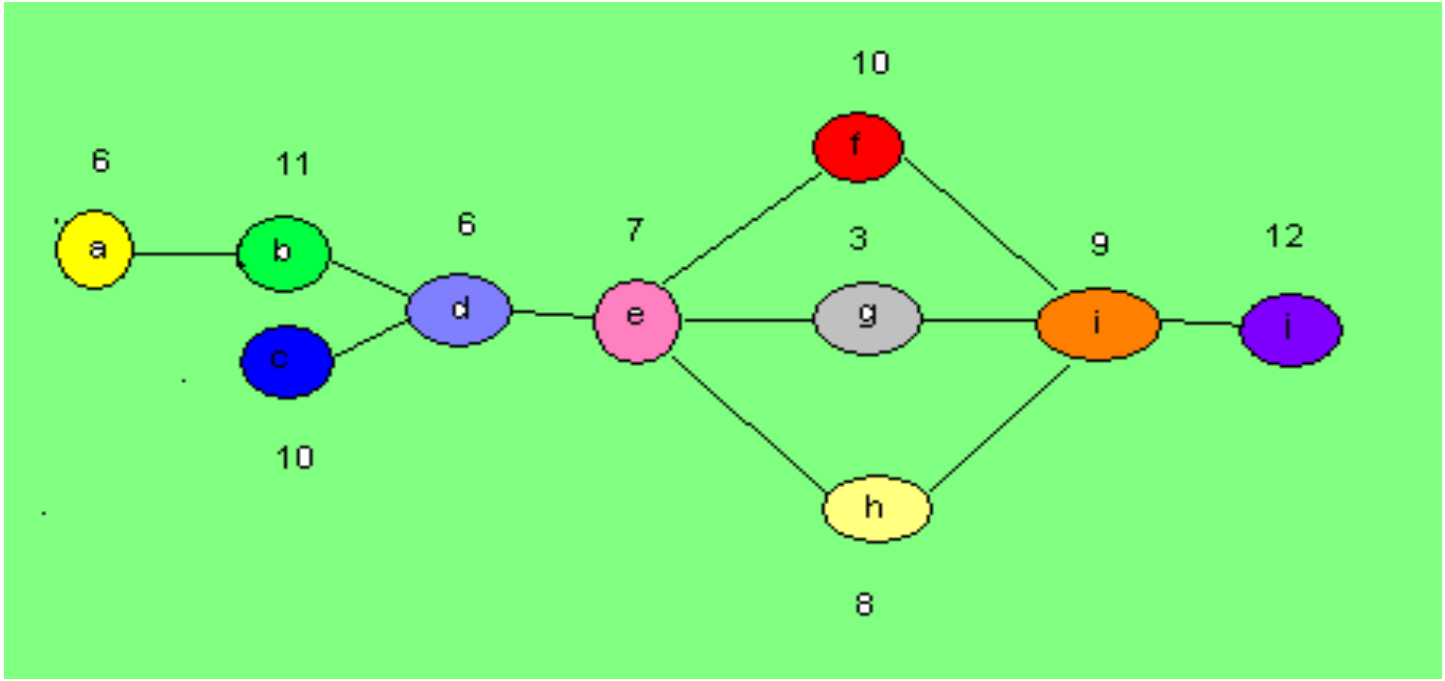
BRANCH AND BOUND

Prueba de optimalidad

- El problema termina cuando no existen subproblemas restantes o la solución de apoyo o incumbente actual es óptima. Si no es así se realiza otra iteración. (Si no existe una solución de apoyo, la conclusión es que el problema no tiene soluciones factibles).

DESARROLLO DE LA INVESTIGACIÒN

Diagrama de precedencias



DESARROLLO DE LA INVESTIGACIÒN

RESUMEN SALBP-1 CON COMSOAL

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	a	6	6	4
	b	11	17	
2	c	10	10	5
	d	6	16	
3	e	7	7	3
	h	8	15	
	g	3	18	
4	f	10	10	2
	i	9	19	
5	j	12	12	9

DESARROLLO DE LA INVESTIGACIÒN

EFICIENCIA:

$E = \sum Tj / (m^* c) = 82 / (5^* 19) = 86.3\%$

TIEMPO MUERTO TOTAL:

$D = (m^*c) - \sum Ti = 23$

DESARROLLO DE LA INVESTIGACIÒN

RESUMEN SALBP-2 CON COMSOAL

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	c	10	10	3
	a	6	16	
2	b	11	11	2
	d	6	17	
3	e	7	7	1
	h	8	15	
	g	3	18	
4	f	10	10	0
	i	9	19	
5	j	12	12	7

DESARROLLO DE LA INVESTIGACIÒN

Tiempo de ciclo del balanceo

C=19

Numero de estaciones

m=5

Eficiencia del balanceo

$E=\sum Ti/ (m*c)$

E=86.3%

Tiempo muerto total o demora del balance

$D= (m*c)- \sum Ti$

D= 13

DESARROLLO DE LA INVESTIGACIÒN

RESUMEN SALBP-1 CON B&B

ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	a	6	6	15
2	b	11	11	0
	c	10	21	
3	d	6	6	1
	e	7	13	
	g	3	20	
4	f	10	10	3
	h	8	18	
5	i	9	9	0
	j	12	21	

DESARROLLO DE LA INVESTIGACIÒN

RESUMEN DEL SALBP-2 CON B&B

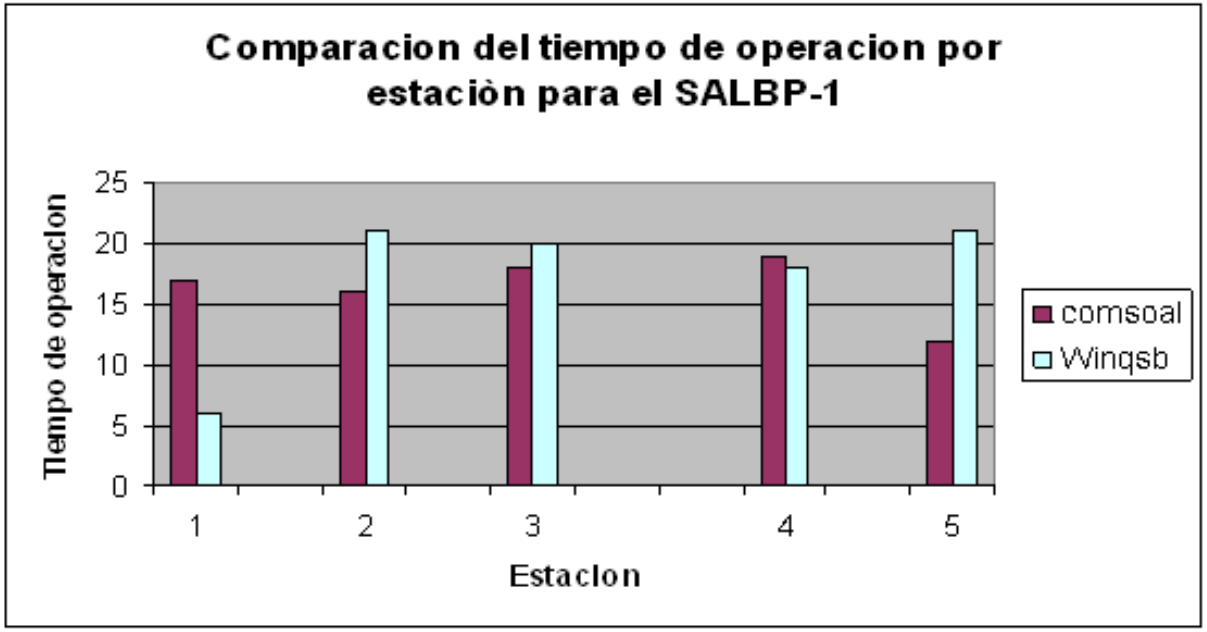
ESTACION	TAREAS	Tj	T acumulado	T ocioso
1	a	6	6	3
	c	10	16	
2	b	11	11	2
	d	6	17	
3	e	7	7	1
	g	3	10	
	h	8	18	
4	f	10	10	0
	i	9	19	
5	j	12	12	7

COMPARACIÒN Y ANALISIS DE RESULTADOS

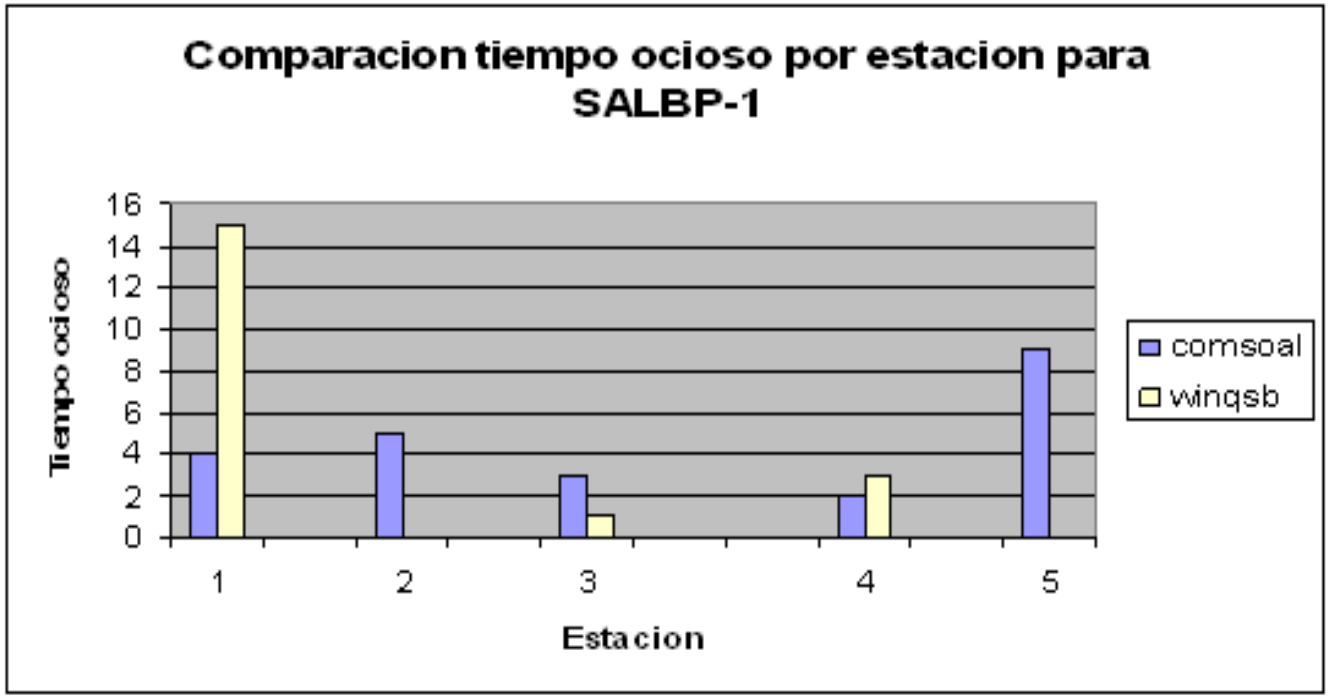
Comparación de resultados SALBP-1

Nº estación	COMSOAL		Branch and Bound(Winqsb)	
	$T_{operación}$	T_{ocioso}	$T_{operación}$	T_{ocioso}
1	17	5	6	15
2	16	4	21	0
3	18	3	20	1
4	19	2	18	3
5	12	9	21	0
Tiempo muerto total	23		19	
Eficiencia real de la línea	86.3%		78.09%	

COMPARACIÒN Y ANALISIS DE RESULTADOS



COMPARACIÒN Y ANALISIS DE RESULTADOS

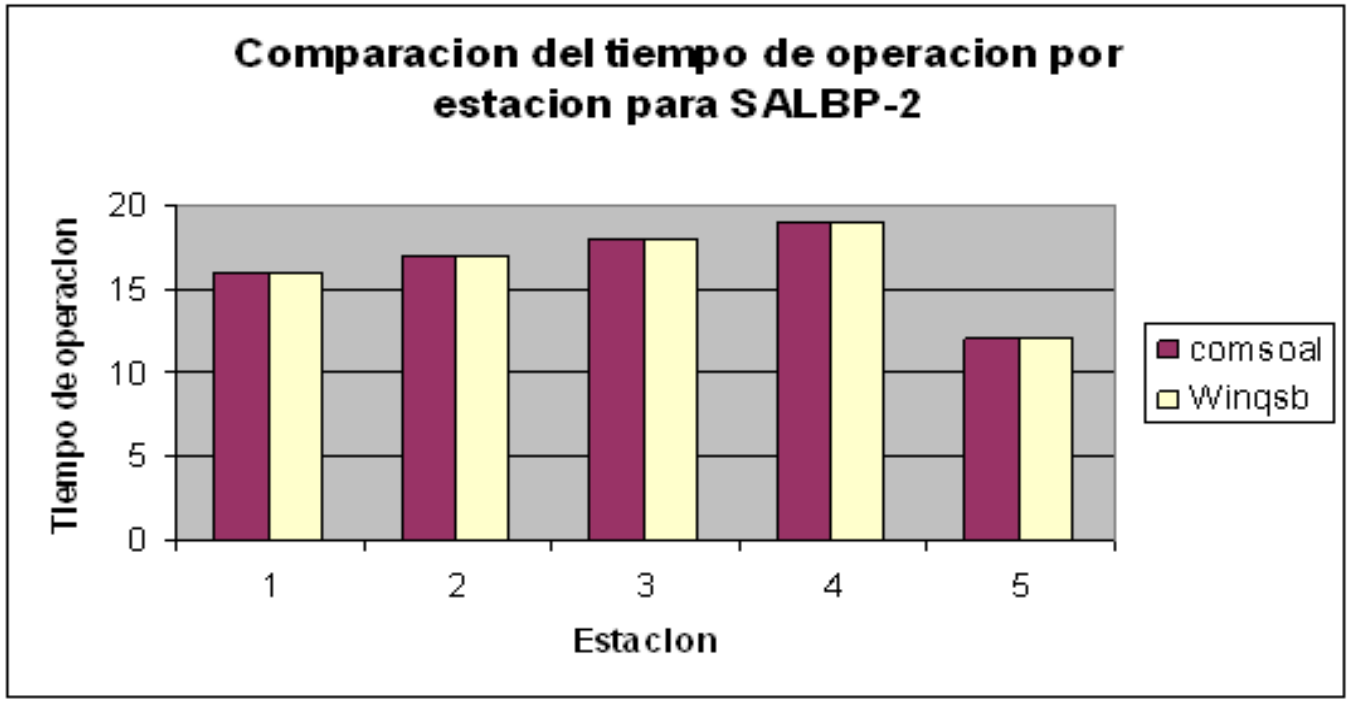


COMPARACIÒN Y ANALISIS DE RESULTADOS

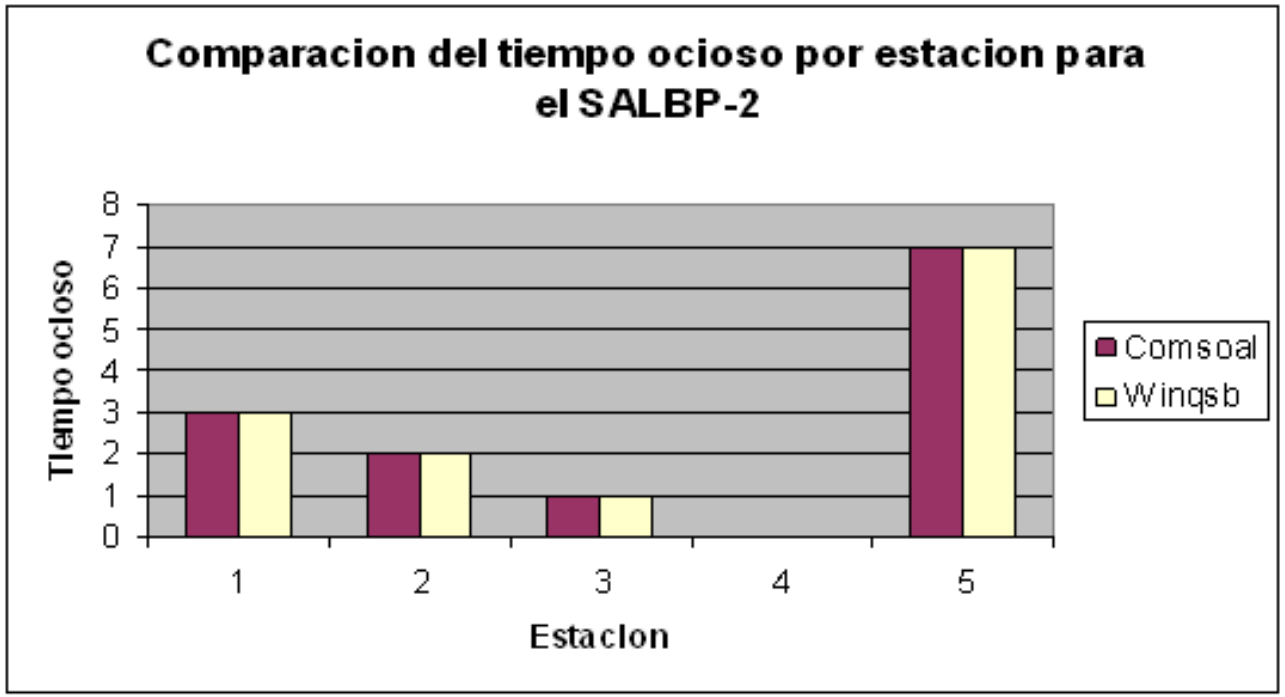
Comparación de resultados SALBP-2

Nº estación	COMSOAL		Branch and Bound(Winqsb)	
	$T_{operación}$	T_{ocioso}	$T_{operación}$	T_{ocioso}
1	16	3	16	3
2	17	2	17	2
3	18	1	18	1
4	19	0	19	0
5	12	7	12	7
Tiempo muerto total	13		13	
Eficiencia real de la línea	86.3%		86.3%	

COMPARACIÒN Y ANALISIS DE RESULTADOS



COMPARACIÒN Y ANALISIS DE RESULTADOS



CONCLUSIONES

- Para el caso SALBP-1, el método comsoal arrojo una mejor eficiencia dado que su tiempo de ciclo fue menor.
- En el SALBP-1 con Comsoal las tareas fueron mejor distribuidas en cada una de las estaciones.
- El Branch and Bound arrojo un valor de tiempo ocioso menor que el comsoal pero por la distribución de las tareas se va a generar un cuello de botella en la estación dos.
- Para el caso SALBP-2 y para este ejemplo, los dos métodos llegaron a los mismos resultados.
- El B&B es un método que permite llegar a una respuesta óptima pero se vuelve ardua su resolución, en casos donde los problemas posean muchas variables.

BIBLIOGRAFIA

- Arcus, A.L. COMSOAL: A computer method of sequencing operations for assembly lines, International Journal of Production Research 4, 259-277. 1966.
- Bard, J. Assembly line balancing with parallel workstations and dead times. International Journal of Production Research. Vol. 37, No. 4, 721-736. 1999.
- Bartholdi, J.J. Balancing two-sided assembly lines: A case study, International Journal of Production Research 31, 2447-2461. 1993.
- Bautista, J. y Pereira, J. Algoritmos de hormigas para un problema de equilibrado de líneas. V Congreso de Ingeniería de Organización, Valladolid-Burgos. 2003.
- Baybars, I. A survey of exact algorithms for the simple assembly line balancing problem" Management Science, 32, 909-932, 1986.

BIBLIOGRAFIA

- Arcus, A.L. COMSOAL: A computer method of sequencing operations for assembly lines, International Journal of Production Research 4, 259-277. 1966.
- Bard, J. Assembly line balancing with parallel workstations and dead times. International Journal of Production Research. Vol. 37, No. 4, 721-736. 1999.
- Bartholdi, J.J. Balancing two-sided assembly lines: A case study, International Journal of Production Research 31, 2447-2461. 1993.
- Bautista, J. y Pereira, J. Algoritmos de hormigas para un problema de equilibrado de líneas. V Congreso de Ingeniería de Organización, Valladolid-Burgos. 2003.
- Baybars, I. A survey of exact algorithms for the simple assembly line balancing problem" Management Science, 32, 909-932, 1986.