Programación de trabajos en líneas de producción en una viña

Franco Basso
Centro de Modelamiento Matemático
Universidad de Chile

SolMat Consultores
fbasso@solmat.cl

30 de julio de 2012

RESUMEN

El presente trabajo resuelve el problema de envasado y embotellamiento de pedidos en líneas de producción en una viña Chilena. El problema es de tipo *scheduling* con características propias. La resolución del problema se aborda desde dos ángulos. El primero consiste en plantear un problema de programación lineal mixto. Los resultados de esta primera estrategia satisfacen los requerimientos de la empresa, sin embargo los altos tiempos computacionales impiden su utilización para casos reales. El segundo enfoque consiste en la utilización de una heurística utilizando *Constraint Programming*. En este caso, los tiempos computacionales son excelentes incluso para casos de gran tamaño. Sin embargo existe un porcentaje de entre el 15 % y el 20 % de los casos estudiados en los cuales el algoritmo no encuentra solución.

PALABRAS CLAVES. MIP, Supply chain, Constraint Programming, Línea de producción
Areas principales. IND – OR in Industry, PM - Mathematical Programming

ABSTRACT

We solve the scheduling problem of a Chilean wineyard, that has to assign packing requests from clients to production lines. The problem has speficic features such as diferent type of wine resources and enological processes contraints. To solve the problem we use two alternative approaches. The first follows a mixed integer optimization problem, which results in a strategy that satisfies the wineyard's requirement but at the cost of a rather large computational time, which precludes its utilization in real instances. The second approach is heuristical and use Contraint Programming to find a feasible solution. In this case, the computational time is reasonable even for real instances. Depending on the data the heuristic approach fails to provide a feasible solution in about 15 % to 20 % of the instances tried.

KEYWORDS. MIP, Supply chain, Constraint Programming, Production Line
Main area. IND – OR in Industry, PM - Mathematical Programming

1. Introducción y definición del problema

En muchas industrias productivas aparece el problema de planificación de la producción de productos en línea. Un ejemplo relevante en Chile es la industria vitivinícola, la cual está sometida a una fuerte competencia a nivel mundial. Una gran parte de la producción de vinos y derivados chilenos son enviados al exterior, representando una parte muy importante de las exportaciones del país. En ese sentido no sólo la calidad del producto importa, puesto que también los clientes exigen prontitud en la entrega y cumplimiento de los plazos. Una parte muy relevante del proceso en esta industria, así como en otras similares (producción de productos alimenticios envasados en general) es la que corresponde al embotellado o envasado y a la fabricación de los lotes del producto para la entrega al cliente.

En este trabajo plantearemos un modelo matemático que represente el problema del embotellado o envasado como un modelo decisional, en el cual el objetivo es ordenar en el tiempo una secuencia de trabajos elementales, que forman parte de una orden o pedido de un cliente. Por ejemplo, un cierto número de botellas de vino de ciertas cepas específicas puede corresponder a un trabajo. Un conjunto de trabajos distintos para un mismo cliente correspondería a una orden. Cada uno de estos trabajos está caracterizado por un vector de atributos, tales como tipo de producto (tipo de cepa, en el caso de los vinos), cantidad de unidades, tipo de envase o botella, tipo de etiquetado, fecha límite de disponibilidad de los insumos para el trabajo, fecha límite de entrega, duración del trabajo (la cual puede depender del equipo o linea de envasado en la cual se realiza), etc.
También consideraremos en el modelo la noción de línea de producción, que corresponde al equipo utilizado para realizar el trabajo (habitualmente, es una línea o correa sin fin, por donde circulan las unidades a envasar). Como normalmente hay varias líneas disponibles, también la asignación de un trabajo a una línea específica forma parte de la decisión del modelo. Por otra parte, la ubicación de un trabajo en el tiempo (secuencia) no es indiferente para el cumplimiento del objetivo de minimizar los tiempos de entrega. Es decir, entre dos trabajos sucesivos en general hay un tiempo dedicado a una proceso denominado *set-up*, que corresponde a las actividades necesarias para preparar la maquinaria para el trabajo i, cuando justo antes se ha realizado el trabajo $i-1$. De manera más precisa, en el caso del envasado de vinos, se necesita lavar los ductos o mangueras por donde ha circulado el producto, de manera de evitar la contaminación entre productos alterando su pureza. Esto puede ocurrir con cualquier producto alimenticio en proceso de envasado. Los tiempos de *set-up* son entonces parte relevante de los datos de entrada. La función objetivo estará generalmente ligada a la minimización del tiempo total de cumplimiento de un conjunto de ordenes o demandas.

2. Solución exacta: Formulación de un problema de programación lineal mixta

La solución propuesta consiste en desarrollar un modelo de optimización lineal mixto que resuelva el problema. Para ello nos valdremos esencialmente de dos tipos de variables. En un primer lugar consideraremos varibles binarias decisionales: Se debe decidir a cual línea de producción y a cual entrega (insumo de vino) se asocia cada trabajo. El segundo tipo de variable corresponde a variables naturales: Se debe decidir en que instante se comienza a producir el trabajo. A continuación presentamos la formulación matemática del modelo.

2.1. Hipótesis de Trabajo

- Se consideran varios tipos de productos.
- Los tiempos de setup sólo dependen del trabajo inicial, final y de la línea en que se hace el trabajo.
- Los tiempos para cada trabajo solo dependen de la línea.

2.2. Conjuntos de Interés

- K: Conjunto de órdenes o clientes.
- N: Conjunto de trabajos individuales e indivisibles.
- P: Conjunto de productos o tipos de vinos.
- L: Conjunto de líneas.
- E: Conjunto de entregas o recursos disponibles.

2.3. Datos de entrada

2.3.1. Constantes

- M: Tiempo máximo de planificación.
- $setup_{n,n',l}$: Tiempo de setup necesario para pasar del trabajo n al trabajo n' en la línea l.

2.3.2. Trabajos

- $tin_{n,p} \in \{0,1\}$: Toma el valor 1 si el trabajo n es del producto p y 0 si no.
- $h_n \in \mathbb{N}$: Tiempo a partir del cual están disponibles los insumos para procesar el trabajo n.
- $rc_{l,n} \in \mathbb{N}$: Tiempo que se demoraría el trabajo n si es procesado en la línea l.
- $lt_n \in \mathbb{N}$: Cantidad de litros del trabajo n.

2.3.3. Órdenes

- $I_k \subseteq N$: Trabajos asociados a la orden k.
- $t_k \in \mathbb{N}$: Tiempo en el cual es requerido que esté lista la orden k.

2.3.4. Entregas

- $ti_e \in \mathbb{N}$: Tiempo inicial de la entrega e.
- $tf_e \in \mathbb{N}$: Tiempo final de la entrega e.
- $vol_e \in \mathbb{N}$: Cantidad de litros de la entrega e.
- $tinto_{e,p} \in \{0,1\}$: Toma el valor 1 ssi la entrega e es del producto p.

2.4. Variables

- $g_n \in \mathbb{N}$: Tiempo en el que se empieza a procesar trabajo n.
- $x_{n,l} \in \{0,1\}$: Toma el valor 1 ssi el trabajo n se precesa en la línea l.
- $y_{n,n'} \in \{0,1\}$: Toma el valor 1 ssi el trabajo n se procesa después que el trabajo n' en la misma línea.
- $yj_{n,n'} \in \{0,1\}$: Toma el valor 1 ssi el trabajo n se procesa justo despues del trabajo n' en la misma linea
- $zb_{n,e} \in \{0,1\}$: Toma el valor 1 ssi el trabajo n se realiza con la entrega e.
- $o_n \in \mathbb{N}$: Tiempo en el que se completa el trabajo n.

2.5. Restricciones

(1)	$\sum_{l \in L} x_{n,l}$	$=$	1	$\forall n \in N$
(2)	g_n	$\geq$	h_n	$\forall n \in N$
(3)	o_n	$=$	$g_n + \sum_{l \in L} x_{n,l} \cdot rc_{n,l}$	$\forall n \in N$
(4)	$y_{n,n'}$	$\leq$	$1 + \frac{g_n - g_{n'}}{M+1}$	$\forall n, n' \in N, n \neq n'$
(5)	$y_{n,n'}$	$\leq$	$1 - x_{n,l} + x_{n',l}$	$\forall n, n' \in N, n \neq n', \forall l \in L$
(6)	$y_{n,n'}$	$\geq$	$x_{n,l} + x_{n',l} - 2 + \frac{g_n - g_{n'}}{M+1}$	$\forall n, n' \in N, n \neq n', \forall l \in L$
(7)	1	$\geq$	$x_{n,l} + x_{n',l} - y_{n,n'} - y_{n',n}$	$\forall n, n' \in N, n > n'. \forall l \in L$
(8)	$o_{n'}$	$\leq$	$2M(1 - yj_{n,n'}) + 2M(2 - x_{n,l} - x_{n',l}) + g_n - setup_{n',n,l}$	$\forall n, n' \in N, n \neq n', \forall l \in L$
(9)	$\sum_{e \in E} zb_{n,e}$	$=$	1	$\forall n \in N$
(10)	$tin_{n,p} \cdot zb_{n,e}$	$\leq$	$tinto_{e,p}$	$\forall n \in N, \forall e \in E, \forall p \in P$
(11)	$(1 - tin_{n,p}) \cdot zb_{n,e}$	$\leq$	$1 - tinto_{e,p}$	$\forall n \in N, \forall e \in E, \forall p \in P$
(12)	o_n	$\leq$	$tf_e \cdot zb_{n,e} + M(1 - zb_{n,e})$	$\forall n \in N, \forall e \in E$
(13)	$ti_e \cdot zb_{n,e}$	$\leq$	g_n	$\forall n \in N, \forall e \in E$
(14)	$\sum_{n \in N} lt_n \cdot zb_{n,e}$	$\leq$	vol_e	$\forall e \in E$
(15)	o_n	$\leq$	t_k	$\forall n \in I_k$
(16)	$yj_{n',n}$	$\leq$	$y_{n',n}$	$\forall n, n' \in N, n \neq n'$
(17)	$yj_{n,n'}$	$\leq$	$2 - y_{n,n''} - y_{n'',n'}$	$\forall n, n', n'' \in N, n \neq n' \neq n''$
(18)	$y_{n,n'}$	$\leq$	$\sum_{n'' \in N} yj_{n,n''}$	$\forall n, n' \in N, n \neq n'$
(19)	$yj_{n,n}$	$=$	0	$\forall n \in N$

Análisis de las restricciones

A continuación se analizarán las restricciones del modelo.

(1) Un trabajo se asigna a una y solo una línea.

(2) Para empezar a realizar un trabajo deben estar los insumos correspondientes.

(3) Definición de o_n.

(4) Definición de $y_{n,n'}$.

(5) Definición de $y_{n,n'}$.

(6) Definición de $y_{n,n'}$.

(7) Definición de $y_{n,n'}$.

(8) Restricción por tiempos de *set up*.

(9) Un trabajo se asigna a una y sola una entrega.

(10) Una entrega se puede solo asociar a un trabajo del mismo producto.

(11) Una entrega se puede solo asociar a un trabajo del mismo producto.

(12) Si un trabajo se asocia a una entrega, el trabajo debe terminar antes del final de la entrega.

(13) Si un trabajo se asocia a una entrega, el trabajo debe empezar después del inicio de la entrega.

(14) La suma de los volúmenes de los trabajos asociados a una entrega no puede sobrepasar su capacidad.

(15) Los trabajos deben estar finalizados antes que expire la fecha de despacho de la orden a la cual pertenecen.

(16) Definición de $yj_{n,n'}$.

(17) Definición de $yj_{n,n'}$.

(18) Definición de $yj_{n,n'}$.

(19) Definición de $yj_{n,n'}$.

2.6. Función Objetivo

$$\min \sum_{k \in K} w_k f_k + \frac{1}{|N|} \sum_{n \in N} o_n$$

En otras palabras se busca minimizar el promedio simple de los tiempos de finalización de cada trabajo sumado a un promedio ponderado de los tiempos de finalización de las ordenes hechas por los clientes.

Los resultados del modelo han resultado satisfactorios, pues han mostrado un alto grado de congruencia con respecto a lo esperado. Sin embargo se ha notado que el tiempo computacional pareciera ser exponencial respecto al número de trabajos. Una heurística parece entonces una salida razonable.

3. Solución Aproximada: Algoritmo Glotón Usando Constraint Programming

En esta sección se presenta un método para encontrar una solución factible al modelo expuesto en 2.5 lo que en muchos casos puede satisfacer los requerimientos de la viña. Para ello, nos valdremos del *Contraint Programming* y más específicamente del software *Comet*

3.1. Sobre Contraint Programming

El *Contraint Programming* es un poderoso paradigma para resolver problemas de búsqueda combinatoriales, basado en una gran variedad de técnicas distintas, provenientes de diversas áreas como la computación, la programación matemática o la investigación de operaciones. La noción de *Contraint Programming* como término unificador de todas estas técnicas aparece durante la decada de 1980 a pesar que los conceptos de restricción o *constraint* y programación ya eran conocidos desde hace décadas. No se trata, por ende, de un área totalmente nueva, pues la mayoría de los problemas - y a veces soluciones- tratados por el *Contraint Programming* son conocidos hace tiempo. Se trata, más bien, de una lista de herramientas que permiten resolver problemas relativamente específicos, utilizando implementaciones computacionales de algoritmos adhoc.

Contraint Programming utiliza restricciones declaradas a través de variables de decisión definidas en conjuntos predeterminadas. La mayoría de las técnicas usadas en *Contraint Programming* suponen que estos conjuntos son discretos (por ejemplo $\mathbb{Z}, \mathbb{N}, \{0, 1\}$), sin embargo existen hoy en día generalizaciones para el caso continuo.

3.2. Pseudocódigo

Algorithm 1 Algoritmo Glotón Usando *Contraint Programming* (**AGUCP**)

1: Ordenar N según urgencia de los trabajos
2: $W \leftarrow N$
3: **while** $W \neq \emptyset$ **do**
4: Seleccionar primer elemento $w \in W$
5: Seleccionar aleatoriamente $l \in L$
6: **try:** $x_{w,l} = 1$
7: **elsetry:** $x_{w,l} = 0$
8: $W \leftarrow W \setminus \{w\}$
9: **end while**
10: $W \leftarrow N$
11: **while** $W \neq \emptyset$ **do**
12: Seleccionar primer elemento $w \in W$
13: $m = 1$
14: **while** TRUE **do**
15: **try:** $g_w = m$, **break**
16: **elsetry:** $g_w > m$, $m \leftarrow m + 1$
17: **end while**
18: $W \leftarrow W \setminus \{w\}$
19: **end while**
20: Seleccionar aleatoriamente una entrega factible para cada trabajo

3.3. Observaciones

- El comando *try* intenta incluir la restricción adicional que le sigue. Si hay infactibilidad incluye la restricción adicional que le sigue al comando *elsetry*.
- El algoritmo es estocástico pues en la línea 5 y 20 se hace una elección al azar. P
- La línea 20 se hace utilizando el comando *label* en el programa *Comet*.

4. Resultados

4.1. Consideraciones generales

En la mayoría de los casos expuestos el algoritmo ha encontrado solución de manera inmediata. A continuación se presenta una tabla de comparación entre el modelo de optimización lineal mixto y el algoritmo glotón usando *contraint programming* (los tiempos se muestran en segundos).

Numero de Trabajos	Tiempo Resolución MIP	Tiempo Resolución AGUCP
10	1 sec	1 sec
12	12 sec	2 sec
15	4 min 34 sec	1 sec
20	4 hrs	1 sec

Observaciones

- Todas los casos se realizaron en un computador con sistema operativo Windows 7 con procesador Inter(R) Core(TM) i5-2430 CPU @ 2.40Ghz y con 8Gb de memoria RAM.
- Para el MIP se utilizó CPLEX 10 mediante el lenguaje GAMS.
- Para el MIP se consideró como tiempo final, el primer momento en el cual se encuentra solución factible.
- Para AGUCP se utilizó el programa *Comet*.
- Del total de casos estudiados hasta el momento, en un 15 % AGUCP no encuentra solución.

4.2. Ejemplo de Solución: $n = 20$

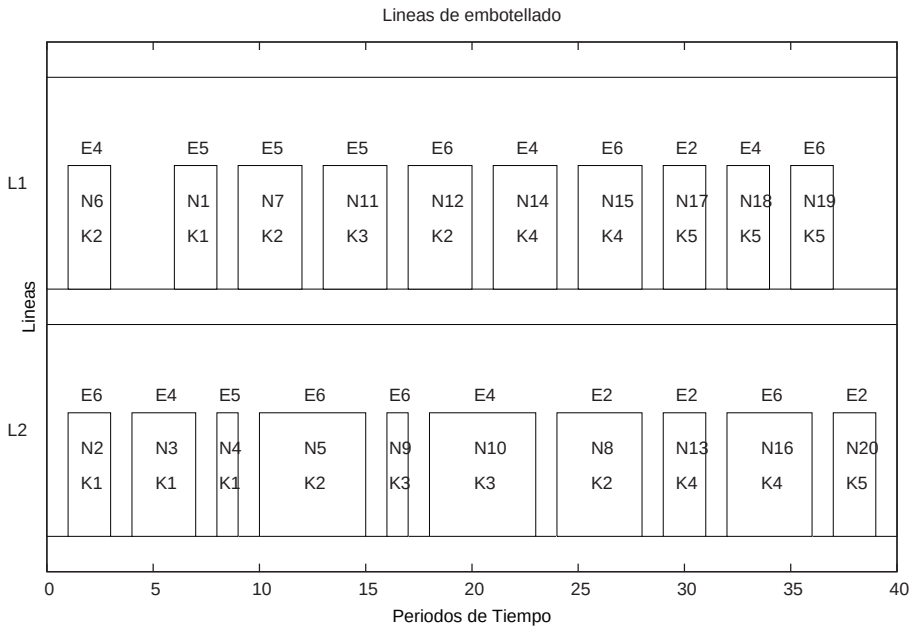

Figura 1: Resultado de AGUCP

5. Conclusiones y trabajos futuros

Los resultados del modelo MIP satisfacen los requerimientos de la viña en el sentido que cumplen todos las restricciones técnicas impuestas por esta. Sin embargo, la gran cantidad de variables, restricciones y su carácter combinatorial hace que sea imposible resolver instancias reales. Por ello en la segunda sección del presente artículo se muestra el *Algoritmo Glotón Usando Constraint Programming*. Dicha heurísitica encuentra solución factible en la gran mayoría de los casos (80-85 %) con una gran velocidad y para instancias reales.
Algunas posibles areas de trabajo futuro son:

- Mejorar el porcentaje de éxito del algoritmo o bien ser capaz de caracterizar los casos en que, existiendo solución, el algoritmo falla.
- Utilizar métodos de mejoramiento de la soluciones encontradas por el algoritmo.
- Incorporar incertidumbre a algunos datos.

Referencias

[1] F. ROSSI, P. VAN BEEK AND T. WALSH, *Handbook of Constraint Programming*

[2] DYNAMIC DECISION TECHNOLOGIES INC, *Comet Tutorial*