MO420 - TP de Geração de Colunas

RA009206 - Luís Guilherme Fernandes Pereira
RA044072 - Igor Ribeiro de Assis

Trabalho 2 - 1o semestre de 2009

1 Introdução

Neste trabalho prático, estudamos o problema de geração de colunas, implementamos essa técnica no resolvedor XPress e usamo-la para resolver o problema de *multi-item capacitated lot sizing* (MILSP).

2 Descrição

Este trabalho consistiu nas seguintes atividades:

1. Implementação de um algoritmo de geração de colunas para resolução do MILSP. Para isso foram necessárias uma política de geração de colunas iniciais e uma regra para geração de colunas posteriores.
2. Geração de limitantes duais a partir da resolução dos problemas mestre restritos.
3. Formulação inteira do ULS, que foi utilizado como problema de pricing.
4. Resolução determinística do ULS, utilizando o algoritmo de programação dinâmica de Wagner e Within.

3 Implementação

A política de geração de colunas iniciais foi da seguinte forma: geramos uma matriz identidade para termos uma base viável, e outros $M * T$ esquemas, um para cada produto em que no esquema t do produto k é produzida toda a demanda dos períodos de $t..T$ e nos períodos de $1..t-1$ é produzida a

demanda do período. Observe que da forma como as colunas iniciais são geradas não necessiariamente se tem uma solução do MILSP[1].

Note que (na maioria dos casos) as colunas da matriz identidade não formam esquemas de produção e portanto não são válidas, por isso a estas colunas são atribuídos custos bastante altos de forma que sejam escolhidas para sair da base conforme colunas vão sendo geradas pelo *pricing*. Vale dizer também que mesmo as colunas geradas pelo pricing podem não ser válidas já que estas são soluções do ULS e podem violar a capacidade de produção C_t da máquina.

Geramos limitantes duais quando o valor da solução do problema de pricing é igual a zero, isto é, não existe coluna que possa entrar na base e portanto chegamos ao ótimo do problema mestre relaxado. Também, a cada solução inteira, temos limitantes primais.

A formulação inteira para o ULS foi da forma:

$$
\begin{aligned}
min \quad & \sum_{t=1}^{n} p_t x_t + \sum_{t=1}^{n} h_t s_t + \sum_{t=1}^{n} f_t y_t & \\
s.a \quad & s_{t-1} + x_t = d_t + s_t & \forall t \in \{1..n\} \\
& x_t \leq M y_t & \forall t \in \{1..n\} \\
& s_0 = 0, s_t, x_t \geq 0, y_t \in \{0,1\} &
\end{aligned}
$$

A formulação utilizada não é a melhor no sentido de que há formulações compactas do ULS que descrevem a envoltória convexa deste problema [Wol00], no entanto a escolha desta formulação se deve ao fato de ser mais direta a transformação de sua solução em planos de produção (colunas) da formulação do MILSP utilizada.

Para resolver o ULS foi utilizada o resolvedor de MIP do XPress, aqui é um ponto que seria vantajoso utilizar a formulação que descreve a envoltória convexa pois poderia ser utilizado um resolvedor de programação linear (Simplex) mais rápido que um de MIP já que na maioria dos casos os resolvedors de PLI utilizam algoritmos como o Simplex em subrotinas que calculam limitantes.

Como se conhece uma formulação compacta que descreve a envoltória convexa do ULS é de se esperar que se tenham algoritmos combinatórios para resolvê-lo, que é o caso. O algoritmo de Wagner e Within foi implementado em $O(n^3)$, utilizando ideias presentes no algoritmo de Floyd-Warshall para grafos.

[1] As colunas (por produto) formam uma matriz triangular superior, portanto a chance de conflitos grande.

Preocupamo-nos em escrever um código que fosse reutilizável, isto é, dado um novo par "problema mestre"-"problema restrito", basta implementar duas classes em C++ segundo certa interface padrão, e poderemos utilizar nosso código sem outras alterações para fazer a geração de colunas via XPress.

4 Análise de resultados

Referências

[Wol00] L.A. Wolsey. *Integer programming*. Wiley Interscience Publications, 2000.