-

MO420A/MC908A: Programação Linear Inteira
Prof. Cid Carvalho de Souza[1] – IC/UNICAMP
2º Trabalho Prático

1 Geração de colunas para o problema de planejamento de produção com múltiplos itens

Neste trabalho será implementado um algoritmo de geração de colunas para a resolução de um problema de planejamento de produção que ocorre em muitos sistemas industriais.
Planejamento de produção é o nome que se dá ao processo de determinar a quantidade de produção nos próximos períodos, durante um intervalo de tempo chamado horizonte de planejamento. A tomada de deciões envolve vários dados, como por exemplo, a demanda por cada item, a capacidade de produção, o custo para armazenamento de produção excedente, o custo de preparação (setup) da máquina, entre outros. Cada um desses dados podem ser fixos ou variar para cada item ou período.
Os sistemas industriais são complexos e deram origem a diferentes problemas de planejamento de produção, tanto que a literatura sobre o assunto é bastante vasta. Uma revisão sobre o tema pode ser vista em [1] e uma classificação dos problemas em [3].
Nós consideramos o problema chamado **"multi-item capacitated lot-sizing problem"** (MILSP), em que deseja-se planejar a produção para uma única máquina ao longo de cada um dos T períodos do horizonte de planejamento de forma a atender as demandas conhecidades a priori de cada um dos M itens em cada período. O planejamento deve minimizar os seguintes custos operacionais:

- f_{it}: custo de setup da máquina para o item i ser produzido no período t;
- h_{it}: custo de armazenamento por uma unidade do item i no período t;
- p_{it}: custo de produção por uma unidade do item i no período t.

[1]Trabalho preparado em colaboração com Edna Hoshino (DCT-UFMS), doutoranda do IC/UNICAMP

No sistema considerado, a máquina é capaz de produzir apenas um tipo de item em cada período, além de possuir uma capacidade máxima de produção em cada período t, denotado por C_t.
Um **esquema de produção** para um item no horizonte de planejamento é o planejamento individual de produção de um item em uma máquina sem restrições de capacidade de produção. Determinar um esquema de produção de custo mínimo é um caso especial de planejamento de produção conhecido como "uncapacitated lot-sizing problem". Um esquema de produção para um item k pode ser representado por um vetor de T triplas (x_{kt}, s_{kt}, y_{kt}) em que x_{kt} representa a quantidade de produção do item no período t, s_{kt} a quantidade de itens que devem ser armazenados no fim do período t e y_{kt} indica se houve ou não produção do item no período t. Todo esquema de produção deve satisfazer:

$$x_{kt} + s_{kt-1} = d_{kt} + s_{kt}, \forall t \in [1..T].$$

Dois esquemas de produção são **conflitantes** se ambos têm produção de itens em um mesmo período, ou seja, se $y_{it} = y_{jt}$, para algum período t. Caso contrário, são ditos não conflitantes.
Uma solução do MILSP consiste em M esquemas de produção não conflitantes, um para cada item, que satisfaçam as restrições de capacidade de produção em cada período.
O MILSP pode então ser definido da seguinte forma:

> **Instância**: um conjunto de M itens, um horizonte de T períodos, demanda d_{kt} para cada item k em cada período t, capacidade de produção C_t para cada período t e custos de produção p_{kt}, de armazenamento h_{kt} e de setup f_{kt}, para cada item k e período t.
>
> **Problema**: encontrar M esquemas de produção, um para cada item tal que:
>
> 1. os esquemas de produção sejam dois a dois não conflitantes;
> 2. a produção em cada período t não ultrapasse a capacidade de produção C_t em cada esquema de produção;
> 3. $\sum_{k=1}^{M} \sum_{t=1}^{T} p_{kt}x_{kt} + h_{kt}s_{kt} + f_{kt}y_{kt}$ **é mínima**.

Em outras palavras, o MILSP procura minimizar a soma dos custos de produção, armazenamento e setup.

1.1 Objetivo

O objetivo deste trabalho é implementar um algoritmo de geração de colunas para resolver a relaxação linear de um modelo de programação linear inteira (PLI) para o MILSP. Recomenda-se que esta implementação seja feita usando o `XPRESS` que é um resolvedor comercial de PLI disponível em algumas máquinas dos laboratórios do `IC`.

1.2 Formulação PLI do MILSP

Seja P_k o conjunto de todos os esquemas de produção para o item k. Para cada esquema p em P_k associamos uma variável binária λ_p, que assume valor 1 se e, somente, se o esquema de produção p é escolhido para compor a solução ótima, e um custo $c_p = \sum_t (p_{kt} x^p_{kt} + h_{kt} s^p_{kt} + f_{kt} y^p_{kt})$, onde (x^p, s^p, y^p) são os vetores de triplas representando o esquema de produção p.
A seguinte formulação de programação linear inteira é um modelo para o MILSP:

$$\max \quad \sum_{k=1}^{M} \sum_{p \in P_k} c_p \lambda_p \tag{1}$$

$$\text{sujeito a} \quad \sum_{k=1}^{M} \sum_{p \in P_k} y^p_{kt} \lambda_p \leq 1, \quad t \in [1..T] \tag{2}$$

$$\sum_{k=1}^{M} \sum_{p \in P_k} x^p_{kt} \lambda_p \leq C_t, \quad t \in [1..T] \tag{3}$$

$$\sum_{p \in P_k} \lambda_p = 1, \quad k \in [1..M] \tag{4}$$

$$\lambda_p \in \{0,1\}, \quad p \in P_k, \quad k \in [1..M]. \tag{5}$$

Uma vez que o número de colunas desta formulação do MILSP é exponencial em T, torna-se inviável carregá-la na memória e resolver sua relaxação linear por um algoritmo simplex, por exemplo. Portanto, utilizaremos o método da geração de colunas para resolver a relaxação linear do modelo acima. Como visto em aula, a idéia do método consiste em iterativamente resolver um problema mestre restrito, que toma apenas um subconjunto das colunas do modelo. Novas colunas vão sendo adicionadas ao modelo sempre que o valor ótimo do subproblema de *pricing* (problema escravo) for negativo.

Note que, o subproblema de *pricing* do modelo do MILSP consiste no problema uncapacitated lot-sizing problem (ULS), que é bastante estudado na litetatura. Existem vários algoritmos para ULS, um dos quais é um algoritmo de programação dinâmica clássico de Wagner e Whitin (veja [2]).

1.3 Implementação

As seguintes tarefas de programação devem ser realizadas:

I1: Usando as bibliotecas providas pelo `XPRESS`[2], implemente um algoritmo de geração de colunas para o MILSP que usa a formulação do problema mestre descrita na subseção 1.2.

I2: Implemente uma função que calcula o limitante dual discutido em sala após a otimização de cada relaxação linear do problema mestre restrito.

I3: Implemente uma subrotina que resolve exatamente o subproblema de *pricing* usando o `XPRESS`. Note que isto implica que você deve formular o ULS usando PLI !

I4: Implemente o algoritmo de programação dinâmica de Wagner e Whitin para o ULS e utilize-o para resolver o subproblema de *pricing*.

1.4 Testes

As instâncias de teste serão disponibilizadas oportunamente na página da disciplina. Elas são baseadas nas instâncias de teste disponibilizadas em `http://www.suerie.de/`.
Faça os experimentos listados a seguir usando estas instâncias.

T1: *(60% da nota)* Execute o código de geração de colunas contendo as implementações dos itens I1, I2 e I3 acima. Note que neste caso, só haverá um limitante primal se, no correr das iterações, a solução ótima do problema mestre restrito for inteira !

T2: *(20% da nota)* Além do que está especificado no item T1, reexecute seu código desta vez usando o algoritmo do item I4 da subseção anterior para resolver o subproblema de *pricing*.

(20% da nota) Reporte e analise os resultados dos testes T1 e T2, utilizando gráficos e tabelas para argumentar suas conclusões.
Obs.: A entrega do relatório é obrigatória !

[2]Ou por qualquer outro resolvedor de sua preferência ao qual você tenha accesso.

1.5 Observações importantes

Para entregar o seu trabalho corretamente, observe os itens listados abaixo:

- o trabalho deve ser feito por grupos de **até** dois alunos.
- a programação deve ser feita em linguagem `C` ou `C++` e compilável em uma instalação Linux padrão (com `gcc` e `g++`).
- o relatório não pode ultrapassar 10 páginas (limite rígido).
- o trabalho deve ser entregue em um arquivo formato `tgz` enviado por *email* para o docente até as 24:00 horas da data fixada para a entrega na página da disciplina. Ao ser descompactado, este arquivo deve criar **no diretório corrente** o diretório `RAxxxxxx<-RAyyyyyy>` contendo os subdiretórios `Programas`, `Texto` e `Testes`.

  No subdiretório `Programas` deverão estar todos os programas fonte e um arquivo `makefile` que compile todo código. Se os programas não compilarem a nota do trabalho será *ZERO*.

  No subdiretório `Texto` deverá estar o arquivo com o seu relatório em formato `ps` ou `pdf`. Se o arquivo não estiver em um destes formatos, a nota do trabalho será *ZERO*.
- Para cada instância, o algoritmo de geração de colunas deverá ser executado até que: *(i)* não haja mais colunas com custo reduzido negativo, ou *(ii)* um tempo **máximo** de 20 minutos tenha sido atingido para aquela instância, ou *(iii)* um total de 2000 colunas tenham sido geradas. O resultado da execução em cada instância deve ser armazenado no subdiretório `Testes`, junto com a melhor solução inteira encontrada em cada teste.

  Nota: *estes parâmetros poderão vir a ser modificados. Caso isto ocorra, você será notificado em aula e na página da disciplina.*
- Deixe claramente explicitado no seu relatório qual foi o conjunto inicial de colunas usado pelo algoritmo.
- As notas serão comparativas entre os diferentes trabalhos entregues. Assim, a qualidade dos relatórios será determinante para obter uma maior ou menor nota. Testes adicionais (por exemplo, comparando o desempenho do algoritmo quando uma única coluna é acrescentada por vez e quando múltiplas colunas são acrescentadas) são bem-vindos. O mesmo vale para gráficos (por exemplo aqueles que ilustrem a evolução típica dos limitantes ao longo das iterações).

Referências

[1] Céline Gicquel, Michel Minoux, and Yves Dallery. Capacitated lot sizing models: a literature review, 2008.

[2] Laurence A. Wolsey. *Integer Programming.* Wiley-Interscience, 1998.

[3] Laurence A. Wolsey. Solving multi-item lot-sizing problems with an mip solver using classification and reformulation. *Manage. Sci.*, 48(12):1587–1602, 2002.