PROPOSTA DE PROBLEMAS

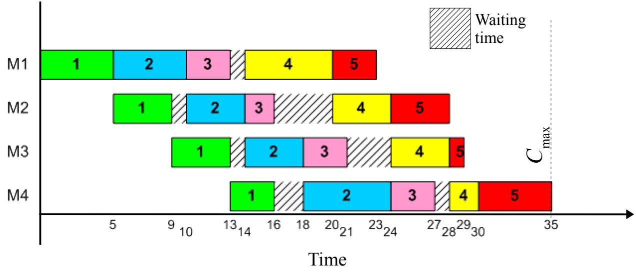
Alberto Francisco Kummer Neto

INF05010 - Otimização Combinatória — Maio, 2019

Problemas propostos

1. PFSP (Permutational Flowshop Scheduling Problem)
2. PMSP (Parallel Machines Scheduling Problem)
3. TSP-DL (Traveling Salesman Problem with Draft Limits)

PFSP
Permutational Flowshop Scheduling Problem



(Fonte: Adaptado de StackOverflow)

Conjuntos e parâmetros:

- $N = \{1, \ldots, \mathrm{n}\}$: conjunto de tarefas a serem processadas
- $M = \{1, \ldots, \mathrm{m}\}$: conjunto de máquinas disponíveis
- Tempos de processamento $\mathrm{T}_{ir} \geqslant 0$, $\forall i \in N$, $r \in M$

Restrições:

- Todas as máquinas devem processar as tarefas em uma mesma sequência (1-2-3-4 e 3-1-2-4 são dois exemplos de sequências)
- As tarefas devem ser processadas na ordem crescente do índice das máquinas (começa na máquina 1, segue para a 2, até chegar na máquina m)
- Processamento sem preempção

Objetivo: Minimizar o *makespan* da máquina m

Solução: Ordem execução das tarefas

- $C_{ri} \geqslant 0$: Variáveis reais que indicam o tempo em que a tarefa $i \in N$ termina de ser processada na máquina $r \in M$
- $C_{\max} \geqslant 0$: Variável real que indicam o tempo final de processamento da última máquina, m
- $D_{ik} \in \{0,1\}$: Variáveis binárias que indicam se a tarefa $i \in N$ precede a tarefa $k \in N$, e só considera $i < k$

$$\text{Minimiza } C_{max} \tag{1}$$

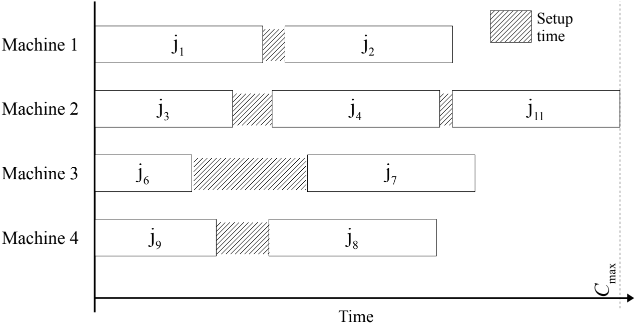
Sujeito a:

$$
\begin{aligned}
& C_{1i} \geqslant \mathrm{T}_{1i} && 1 \leqslant i \leqslant \mathrm{n} && (2)\\
& C_{ri} - C_{r-1,i} \geqslant \mathrm{T}_{ri} && 1 \leqslant i \leqslant \mathrm{n}, 2 \leqslant r \leqslant \mathrm{m} && (3)\\
& C_{ri} - C_{rk} + \mathrm{P} \cdot D_{ik} \geqslant \mathrm{T}_{ri} && 1 \leqslant i < k \leqslant \mathrm{n}, 1 \leqslant r \leqslant \mathrm{m} && (4)\\
& C_{ri} - C_{rk} + \mathrm{P} \cdot D_{ik} \leqslant \mathrm{P} - \mathrm{T}_{rk} && 1 \leqslant i < k \leqslant \mathrm{n}, 1 \leqslant r \leqslant \mathrm{m} && (5)\\
& C_{max} \geqslant C_{\mathrm{m}i} && 1 \leqslant i \leqslant \mathrm{n} && (6)\\
& C_{ri} \geqslant 0 && 1 \leqslant r \leqslant \mathrm{m}, 0 \leqslant i \leqslant \mathrm{n} && (7)\\
& D_{ik} \in \{0,1\} && 1 \leqslant i < k \leqslant \mathrm{n} && (8)
\end{aligned}
$$

(Fonte: Tseng et al. (2003))

As restrições (2) garantem que horário final de processamento de cada tarefa da máquina 1 é pelo menos tão grande quanto seu tempo de processamento. As restrições (3) garantem que a diferença entre o tempo que a tarefa completa na máquina na máquina atual, e o tempo que a tarefa completou na máquina antecessora seja tão grande quanto o tempo de processamento da tarefa na máquina atual. As restrições (4 – 5) garantem uma ordem de processamento comum das tarefas a todas as máquinas do problema. Considere P como um valor suficientemente grande (técnica *big M*). As restrições (6) são responsáveis por calcular o tempo final de processamento da última tarefa da última máquina. As restrições (7 – 8) indicam o domínio das variáveis de decisão do problema.

PMSP
Parallel Machines Scheduling Problem



(Fonte: Autoria própria)

Conjuntos e parâmetros:

- $N = \{1, \ldots, n\}$: conjunto de tarefas a serem processadas
- $M = \{1, \ldots, m\}$: conjunto de máquinas disponíveis
- p_{jk}: Tempo de processamento da tarefa $j \in N$ na máquina $k \in M$
- s_{ijk}: tempo de preparo da máquina $k \in M$ para processar a tarefa $j \in N$, tendo a tarefa $i \in N$ como predecessora $(\mathrm{s}_{iik} = 0, \forall k \in M)$
- Tempo total $\mathrm{G}_{ijk} = \mathrm{s}_{ijk} + \mathrm{p}_{jk}$

Restrições:

- Cada tarefa deve ser processada em alguma máquina
- Processamento sem preempção

Objetivo: Minimizar o *makespan* da máquina com tempo de processamento mais longo

Solução: Ordem alocação das tarefas às máquinas

- $x_{ijk} \in \{0, 1\}$: Variáveis binárias que indicam se a tarefa $i \in N$ precede a tarefa $j \in N$ na máquina $k \in M$
- $C_j \geqslant 0$: Variáveis reais que indicam o tempo em que a tarefa $j \in N$ termina de ser processada
- $C_{\max} \geqslant 0$: Variável real que indicam o tempo final de processamento da máquina que operou por mais tempo

$$\text{Minimiza } C_{max} \tag{9}$$

Sujeito a:

$$\sum_{\substack{i=0 \\ i \neq j}}^{\mathrm{n}} \sum_{k=1}^{\mathrm{m}} x_{ijk} = 1 \qquad 1 \leqslant j \leqslant \mathrm{n} \tag{10}$$

$$\sum_{\substack{i=0 \\ i \neq h}}^{\mathrm{n}} x_{ihk} - \sum_{\substack{j=0 \\ j \neq h}}^{\mathrm{n}} x_{hjk} = 0 \qquad 1 \leqslant h \leqslant \mathrm{n}, \; 1 \leqslant k \leqslant \mathrm{m} \tag{11}$$

(Continua no próximo slide)

$$C_j \geqslant C_i + \sum_{k=1}^{\mathrm{m}} \mathrm{G}_{ijk} x_{ijk} + \mathrm{V}\left(\sum_{k=1}^{\mathrm{m}} x_{ijk} - 1\right) \qquad 0 \leqslant i \leqslant \mathrm{n} \tag{12}$$

$$C_{max} \geqslant C_j \qquad 1 \leqslant j \leqslant \mathrm{n} \tag{13}$$

$$\sum_{j=0}^{\mathrm{n}} x_{0jk} = 1 \qquad 1 \leqslant k \leqslant \mathrm{m} \tag{14}$$

$$C_0 = 0 \tag{15}$$

$$C_j \geqslant 0 \qquad 1 \leqslant j \leqslant \mathrm{n} \tag{16}$$

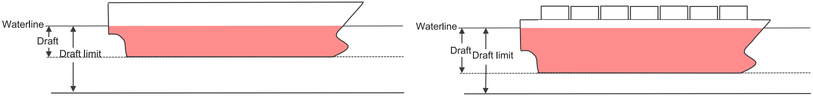
$$x_{ijk} \in \{0, 1\} \qquad \begin{array}{l} 0 \leqslant i \leqslant \mathrm{n}, \\ 0 \leqslant j \leqslant \mathrm{n}, \\ 1 \leqslant k \leqslant \mathrm{n} \end{array} \tag{17}$$

(Fonte: Ezugwu (2019))

Observação: O modelo considera uma tarefa artificial 0, que possui tempo de processamento 0 e exige 0 unidades de tempo de preparo em qualquer máquina. Essa tarefa artificial é utilizada na modelagem das restrições (10 – 12), (14) e (15).

As restrições (10) garantem que cada terefa possui exatamente uma tarefa predecessora em alguma máquina, mesmo que seja a tarefa artifical 0. As restrições (11) modelam a conservação de fluxo das tarefas de uma máquina. As restrições (12) calculam o tempo em cada tarefa termina de ser processada, enquanto as restrições (13) são responsáveis por calcular o tempo final de processamento da máquina que operou por mais tempo. Considere V como um valor suficientemente grande (técnica *big M*). As restrições (14) garantem que a tarefa artificial 0 tem alguma tarefa como sucessora, em todas as máquinas, e a restrição (15) força que a tarefa artificial não ocupe tempo de processamento. Por fim, as restrições (16 – 17) definem os domínios das variáveis de decisão.

TSPDL
Traveling Salesman Problem with Draft Limits



(Fonte: Adaptado de Rakke et al. (2012))

Conjuntos e parâmetros:

- Uma única embarcação
- $V = \{1, \ldots, \mathrm{v}\}$: Conjunto de portos
- Garagem no porto $1 \in V$
- $V \setminus \{1\}$: Conjunto de portos consumidores
- Cada porto $j \in V \setminus \{1\}$ tem uma demanda de $\mathrm{d}_j \geqslant 0$ unidades
- Cada porto $j \in V \setminus \{1\}$ possui um limite de profundidade $\mathrm{l}_j \geqslant 0$
- Garagem com oferta suficientemente grande $(\sum_{j \in V \setminus \{1\}} \mathrm{d}_j)$
- Custo c_{ij} para a embarcação fazer uma viagem entre os portos i e j, para todas os pares de viagens possíveis $(i, j) \in V \times V$

Restrições:

- A embarcação deve sair e retornar ao porto garagem exatamente uma vez
- A embarcação deve visitar cada porto consumidor uma única vez
- A embarcação deve atender totalmente às demandas dos portos durante as visitas
- A porcentagem submersa da embarcação é diretamente proporcional a quantidade de carga que ela carrega
- A embarcação não pode atracar em um porto caso a profundidade do casco extrapole o limite de profundidade do porto

Objetivo: Rota de menor custo (ciclo hamiltoniano)

Solução: Ordem de visitação dos portos

- $x_{ij} \in \{0,1\}$: Variáveis binárias que indica se o atendimento do porto $i \in V$ precede o atendimento do porto $j \in V$
- y_{ij}: Variáveis reais que indicam a quantidade de carga da embarcação durante a viagem entre os portos $(i,j) \in V \times V$

$$\text{Minimiza } \mathrm{c}_{ij} x_{ij} \tag{18}$$

Sujeito a:

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall j \in V \tag{19}$$

$$\sum_{j \in V} x_{ij} = 1 \qquad \forall i \in V \tag{20}$$

$$\sum_{i \in V} y_{ij} - \sum_{i \in V} y_{ji} = \mathrm{d}_j \qquad \forall j \in V \setminus \{1\} \tag{21}$$

(Continua no próximo slide)

$$\sum_{j \in V} y_{1j} = \sum_{i \in V} \mathrm{d}_i \tag{22}$$

$$\sum_{i \in V} y_{i1} = 0 \tag{23}$$

$$0 \leqslant y_{ij} \leqslant \mathrm{l}_j x_{ij} \qquad \forall (i,j) \in V \times V \tag{24}$$

$$x_{ij} \in \{0, 1\} \qquad \forall (i,j) \in V \times V \tag{25}$$

(Fonte: Rakke et al. (2012))

Observação: O modelo considera que o porto garagem possui uma demanda $\mathrm{d}_1 = 0$, e um limite de profundidade suficientemente grande $(\mathrm{l}_1 = \sum_{j \in V \setminus \{1\}} \mathrm{d}_j)$.

As restrições (19) e (20) garantem que cada porto possua um predecessor e um sucessor, respectivamente. As restrições (21) garantem o atendimento da demanda dos portos consumidores, ao mesmo tempo que modela a conservação de fluxo em grafos. As restrições (22) forçam que a embarcação saia devidamente carregada do porto garagem. As restrições (23) garantem que a embarcação só retorne para o porto garagem quando estiver vazia. Por efeito, as restrições (21 – 23) modelam a eliminação de subrotas. As restrições (24) definem os domínios das variáveis de decisão y_{ij}, e relaciona-as às variáveis binárias x_{ij}. Ao mesmo tempo, essas restrições modelam os limites de profundidades dos portos consumidores. Por fim, as restrições (25) definem o domínio das variáveis de decisão x_{ij}.