

Lista de exercícios 1

Nos vamos usar o problema da programação em flow shop permutacional (ingl. permutation flow shop scheduling problem, PFSSP) como exemplo recorrente nas listas nesse semestre. Segue a definição do problema

MINIMIZAR O TEMPO DE TÉRMINO NO FLOW SHOP PERMUTACIONAL

Instância Um conjunto de tarefas $J = [n]$ a serem executadas nas máquinas $M = [m]$. Uma tarefa $j \in J$ possui m operações com tempo de execução p_{ij} para $i \in M$. A i -ésima operação de cada tarefa deve ser executada na máquina i . Em cada instante, uma tarefa pode ser processada por uma única máquina, e cada máquina pode processar no máximo uma tarefa. Uma vez iniciado, o processamento de uma tarefa tem que ser executada sem interrupção (*non-preemptive*). Além disso, as tarefas devem ser processadas em cada máquina na mesma ordem.

Solução Uma permutação π das tarefas que define o escalonamento das tarefas. (A saber: a tarefa π_j termina a execução na máquina i no momento $C_{i,\pi_j} = p_{i,\pi_j} + \max\{C_{i-1,\pi_j}, C_{i,\pi_{j-1}}\}$, com $C_{0j} = C_{i,\pi_0} = 0$.)

Objetivo Minimizar o tempo de término (“makespan”) $C_{\max} = C_{m,\pi_n}$ da última tarefa executada.

Exercício 1 (Vizinhanças, 3 pt)

Considere as seguintes três vizinhanças para o PFSSP, para uma dada permutação π :

- a) A vizinhança “1-flip” que troca duas tarefas adjacentes.
- b) A vizinhança “ k -reverse” que inverte a ordem de $k > 2$ tarefas consecutivas. (Nota que essa vizinhança generaliza a primeira, que é o caso $k = 2$.)
- c) A vizinhança “1-shift” que remove uma tarefa de solução e re-insere em outra posição.

Para todas três vizinhanças, informa se ela é simétrica? Conectada? Fracamente otimamente conectada? Exata?

As respostas tem que ser justificadas (i.e. eles tem que provadas).

Exercício 2 (Busca local para um problema polinomial, 3pt)

Estude a busca local na vizinhança 1-flip no problema 1 || $C_{\max}$. Neste problema temos n tarefas com tempos de processamento p_i , $i \in [n]$ que devem ser executadas em uma única máquina. Uma solução do problema é uma permutação π das tarefas, e o objetivo é encontrar a permutação que minimiza a soma dos tempos de término de todas tarefas $C_{\text{sum}} = \sum_{i \in [n]} C_i$, onde, para uma permutação π , o tempo de término é definido recursivamente por $C_{\pi_1} = p_{\pi_1}$ e $C_{\pi_{i+1}} = C_{\pi_i} + p_{\pi_{i+1}}$ para $i \in [n-1]$.

- a) Compare as estratégias “first improvement” e “best improvement”. Repete o experimento para cada estratégia 100 vezes com instâncias iniciais aleatórias. Gera as instâncias aleatórias da seguinte forma: para cada $n \in \{100, 200, \dots, 500\}$, gera primeiramente os tempos $p_1, \dots, p_n$ tirados de $U\{1, 100\}$. Depois gera 20 permutações aleatórias para cada n . Compare a média aritmética (μ) e desvio padrão (σ) do tempo, número de iterações e função objetivo C_{sum} . Apresenta uma análise dos resultados.
- b) Apresente um algoritmo ótimo exata para o problema. Compara a implementação em termos de tempo e função objetivo com as buscas locais.

Entrega dos resultados: para os dois itens um arquivo texto com os resultados individuais da forma

alg	n	no	time	iterations	value
FI	100	1	17.3	23764	88123
FI	100	2	17.3	23764	88123
BI	100	1	32.2	41234	88123
EX	100	1	32.2	NA	88123
...					

com os seguintes campos:

- “alg”: categorias “FI” para “first improvement” e “BI” para “best improvement” e “EX” para “algoritmo exato”;
- “n”: tamanho da instância $n \in \{100, 200, \dots, 500\}$;
- “no”: número da permutação entre $1, \dots, 20$
- “time”: tempo de execução, em segundos;
- “iterations”: número de iterações (nota que o algoritmo exato não tem iterações: neste caso informar “NA” na tabela);
- “value”: soma dos tempos de término C_{sum}

Exercício 3 (Busca local para um problema NP-completo, 4pt)

Estude a busca local na vizinhança 1-shift no PFSSP. Use as instâncias ta001, ..., ta060. As instâncias e os melhores valores conhecidos podem ser encontrados em <http://www.inf.ufrgs.br/~mrpritt/msc/pfssp.zip>.

- Compare as estratégias “first improvement” e “best improvement”. Repete o experimento para cada estratégia 100 vezes para cada instância. Compare a média aritmética (μ) e desvio padrão (σ) do tempo, número de iterações e de C_{max} das duas estratégias. Apresenta um histograma dos valores de C_{max} para as duas estratégias.
- Compare com uma busca local monótona randomizada. Testa com valores de

$$p \in \{0.0, 0.05, 0.25, 0.5, 0.75, 1.0\}.$$

Repete o experimento 15 vezes para cada instância e valor de p . Relate o tempo até encontrar a solução ótima, ou o melhor valor encontrada com de um tempo limite de 15 segundos e o número de iterações numa tabela da forma $\mu \pm \sigma$.

- Determine a complexidade empírica do algoritmo do tempo para encontrar a solução ótima e do número de iterações para encontrar a solução ótima em função do tamanho da instância. O tamanho da instância neste caso é o número de operações nm .

Observações:

- Para um desempenho razoável considere implementar um cálculo eficiente da função objetivo.
- Para escolher um vizinho melhor ou arbitrário uniformemente sem armazenar todos vizinhos pode-se usar *reservoir sampling*: ao encontrar o i -ésimo vizinho candidato, aceitá-lo com probabilidade $1/i$.

Entrega dos resultados:

- Para o primeiro item um arquivo texto com os resultados individuais da forma

alg	instance	rep	time	iterations	value
FI	ta001	1	17.3	23764	88123
FI	ta001	2	17.3	23764	88123
BI	ta001	1	32.2	41234	88123
...					

com os seguintes campos:

- “alg”: categorias “FI” para “first improvement” e “BI” para “best improvement”;
 - “instance”: nome da instância;
 - “rep”: o número da replicação entre $1, \dots, 100$
 - “time”: tempo de execução, em segundos;
 - “iterations”: número de iterações;
 - “value”: tempo de término $C_{\max}$
- Para o segundo item um arquivo texto com os resultados individuais, da forma

p	instance	rep	time	iterations	value
0.0	ta001	1	30.1	2943	88123
0.0	ta001	2	33.8	1234	88124
...					

com os seguintes campos:

- “p”: probabilidade na BLMR;
- “instance”: nome da instância;
- “rep”: o número da replicação entre $1, \dots, 15$
- “time”: tempo de execução, em segundos;
- “iterations”: número de iterações;
- “value”: tempo de término $C_{\max}$

Data de entrega: 25 de fevereiro de 2022.

Regras para listas de exercícios

1. Os exercícios podem ser resolvidos em colaboração com outros, mas a entrega é individual informando os eventuais colaboradores.
2. A entrega é eletrônica, num único arquivo com todos artefatos (relatório, não escrito a mão, em formato PDF, código fonte, e tabelas).
3. Para receber pontos as respostas devem ser justificadas (i.e. provadas quando não são óbvias).
4. Somente entregem respostas que vocês sabem explicar pessoalmente.