

Universidade Federal da Fronteira Sul
Ciência da Computação

Laboratório de Programação Matemática e Otimização Combinatória

Prof. Andrei Braga

Trabalho 3

Instruções:

1. Prazo de entrega: Consultar a página do trabalho no SIGAA.
2. Forma de entrega: Através do SIGAA, deverá ser enviado o seguinte:
 - (a) um pacote ZIP contendo a implementação pedida no itens 2 a 4 abaixo;
 - (b) um arquivo PDF contendo o seu nome, a sua matrícula e as respostas dos itens 1, 3 e 5 abaixo.
3. Deverão ser seguidas as demais instruções dadas em aula.
4. O plágio e a cola serão tratados de forma rígida: os envolvidos receberão nota zero.

Nota:

A atribuição da nota deste trabalho levará em conta o seguinte:

- Peso 85%: A qualidade da resposta apresentada em si;
- Peso 15%: A qualidade da resposta apresentada em comparação à qualidade das respostas apresentadas pelos demais estudantes.

Neste trabalho, você deve considerar o problema descrito a seguir e fazer o que é pedido abaixo:

Problema de Alocação Generalizado (*Generalized Assignment Problem*). Seja I um conjunto de máquinas e J um conjunto de tarefas. O custo da máquina $i \in I$ processar a tarefa $j \in J$ é dado por c_{ij} e a quantidade de recursos da máquina i consumidos para processar a tarefa j é dado por a_{ij} . A quantidade total de recursos da máquina i é dado por b_i . Uma máquina pode processar múltiplas tarefas, mas uma tarefa deve ser processada por exatamente uma máquina. O problema consiste em alocar as tarefas de J para processamento nas máquinas de I de modo que o custo total seja mínimo.

1. Descrever um modelo de programação linear inteira para o Problema de Alocação Generalizado.
2. Em Python, usando a biblioteca PySCIP0pt [2, 1], implementar um programa para resolver o modelo descrito no item 1. Obrigatoriamente, o seu programa deve executar os comandos a seguir antes de proceder com a resolução do modelo:

```
modelo.setParam("presolving/maxrounds", 0) # modelo é a variável que armazena o modelo
modelo.setParam("separating/maxrounds", 0) # de programação linear inteira
```
3. Descrever restrições válidas que, ao serem adicionadas ao modelo do item 1, resultam em um modelo mais *forte* para o problema.
4. Em Python, usando a biblioteca PySCIP0pt, implementar um programa para resolver o modelo descrito no item 3. Obrigatoriamente, o seu programa deve executar os comandos a seguir antes de proceder com a resolução do modelo:

```
modelo.setParam("presolving/maxrounds", 0) # modelo é a variável que armazena o modelo
modelo.setParam("separating/maxrounds", 0) # de programação linear inteira
```
5. Executar os programas dos itens 2 a 4 para todas as instâncias especificadas para o trabalho – se necessário, você pode interromper a execução dos programas após 5 minutos de tempo corrido (usando o parâmetro `limits/time` da biblioteca PySCIP0pt). Elaborar uma tabela contendo as seguintes informações para cada instância:
 - Para o programa do item 2:
 - O valor da solução encontrada;
 - O número de nós da árvore de enumeração construída;
 - A diferença percentual entre o valor da solução encontrada e o valor ótimo;
 - O tempo de execução;
 - Para o programa do item 4:
 - O valor da solução encontrada;
 - O número de nós da árvore de enumeração construída;
 - A diferença percentual entre o valor da solução encontrada e o valor ótimo;
 - O tempo de execução;
 - O valor ótimo;
 - Outras informações que você considerar relevante.
6. Analisar os resultados obtidos no item anterior respondendo ao seguinte:
 - Na sua interpretação, entre os modelos considerados nos itens 2 e 4, qual modelo apresenta os melhores resultados?

Referências

- [1] PySCIP0pt. *PySCIP0pt*. Disponível em <https://github.com/scipopt/PySCIP0pt>. 2024.
- [2] PySCIP0pt. *PySCIP0pt Documentation*. Disponível em <https://pyscipopt.readthedocs.io/en/latest/>. 2024.