Prova 3 - 21/Novembro/2012

## Questões

**Observação:** Qualquer fraude implicará em nota 0.0 para os envolvidos.

Considere o Problema **ESCAL$_{10}$**: Um computador possui 3 núcleos de processamento (cores). Há um conjunto de $n$ tarefas $1, \ldots, n$ para serem processadas. Para processar uma tarefa $i$, ela deve ser executada de maneira ininterrupta por algum dos 3 núcleos de processamento por tempo $t_i$ (os 3 núcleos de processamento são idênticos). Quando um núcleo estiver processando uma tarefa este não poderá executar outra tarefa até que a tarefa em execução seja totalmente terminada. Cada tarefa deve ser executada em exatamente um núcleo. Um escalonamento de tarefas no computador consiste de uma atribuição de tarefas para os núcleos de processamento. O tempo gasto por um núcleo é a soma dos tempos das tarefas escalonadas/executadas no núcleo. O tempo do escalonamento é o maior tempo gasto por um dos núcleos do computador. O objetivo do problema é encontrar um escalonamento de tempo mínimo.

1. (2.5 pts) Faça uma formulação em programação linear inteira para resolver o problema ESCAL$_{10}$.

2. (2.5 pts) Descreva um algoritmo de aproximação com fator de aproximação estritamente menor que 2 para ESCAL$_{10}$. Justifique a complexidade de tempo e prove o fator de aproximação.

3. (2.5 pts) Projete um algoritmo exato, que utilize um resolvedor por programação linear como subrotina (não é programação linear inteira) para resolver o problema ESCAL$_{10}$ através do método *Branch and Bound*. Seu algoritmo exato pode supor a existência de uma formulação da questão (1) e um algoritmo de aproximação da questão (2). Descreva como você faria a enumeração do seu algoritmo exato, e como ocorre a poda no processo de enumeração.

4. (2.5 pts) Considere o problema **BIN-PACKING**. Temos uma lista de itens $\{1, \ldots, n\}$, cada item $i$ com peso $0 \leq t_i \leq 1$. Deve-se empacotar todos os itens no menor número de recipientes de capacidade 1 (o peso total dos itens empacotados em um recipiente não pode ultrapassar 1). Mostre que o algoritmo First-Fit (FF) descrito abaixo para o problema *bin packing* é tal que para qualquer instância $I$ vale que $\mathrm{FF}(I) \leq 2\mathrm{OPT}(I)$.

Algoritmo FF

1. $k \leftarrow 0$.
2. Para $i \leftarrow 1$ até $n$ faça
3. se existe recipiente $B_j$ com $1 \leq j \leq k$ tal que $i$ cabe em $B_j$
4. então coloque $i$ no primeiro recipiente $B_j$ onde $i$ cabe;
5. senão faça $k \leftarrow k + 1$ e coloque item $i$ em $B_k$.
6. Devolva o empacotamento nos recipientes $B_1, B_2, \ldots, B_k$.