

Trabalho 2

Programações inteiras

Neste trabalho você deverá implementar dois algoritmos diferentes. Cada um deles valerá um total de 5 pontos, que serão avaliados de acordo com (3pts) compreensão do algoritmo (2pt) funcionamento. A linguagem pode ser C, java ou python. Você pode usar bibliotecas de álgebra linear.

- (i) Algoritmo de aproximação primal-dual para o problema de cobertura por conjuntos, visto nas aulas 19 e 20.
- (ii) Método guloso de Ford-Fulkerson - https://pt.wikipedia.org/wiki/Algoritmo_de_Ford-Fulkerson - para achar um fluxo máximo, que você deverá pesquisar por conta própria (e que também será avaliado no 2o exame). Use a versão com depth-first search, que você deve implementar também.

Prazo

1. Entrega: dia 27/06. Não será necessário imprimir o código. Mas você precisará responder perguntas em sala sobre o código. No dia anterior, 26/06, você deverá enviar o código para o meu GMAIL, em um arquivo .TXT. O assunto deve ser PRECISAMENTE

[DCC035 trabalho2] arquivo do TP

2. Entrega em laboratório: entre os dias 26/06 e 29/06.
3. Suponha que seu dia favorito seja segunda-feira, sua segunda opção seja quarta-feira. Mande um email para o meu GMAIL cujo assunto seja PRECISAMENTE

[DCC035 trabalho2] segunda quarta

Entrada

- (i) Cobertura por conjuntos:
 - (a) Matriz N cujas linhas são indexadas por elementos, e colunas por subconjuntos. Uma entrada será 1 se o elemento pertencer ao conjunto, 0 caso contrário.
 - (b) Vetor \mathbf{c} que indica o custo de cada subconjunto.
 - (c) Formato: arquivo txt contendo apenas números, espaços e line-breaks. Primeira linha: quantidade de elementos. Segunda linha: quantidade de subconjuntos. Terceira linha: vetor \mathbf{c} , cujas entradas serão separadas somente por espaços. Linhas seguintes: matriz N , com entradas de cada linha separadas por espaços, e linhas de N separadas por line-breaks.
- (ii) Ford-Fulkerson

- (a) Matriz N , de incidência do grafo dirigido. Assuma que a fonte do fluxo será o primeiro vértice de N , e o sorvedouro será o último vértice.
- (b) Vetor \mathbf{c} , de números inteiros, que indicada a capacidade de cada arco.
- (c) Formato: arquivo txt contendo apenas números, espaços e line-breaks. Primeira linha: quantidade de vértices. Segunda linha: quantidade de arcos. Terceira linha: vetor \mathbf{c} , cujas entradas serão separadas somente por espaços. Linhas seguintes: matriz N , com entradas de cada linha separadas por espaços, e linhas de N separadas por line-breaks.

Saída Em ambos os casos, os algoritmos devem fornecer saídas a cada iteração.

(i) Cobertura por conjuntos:

- (a) Cada iteração do algoritmo produz dois vetores. Um vetor \mathbf{y} , inteiro, de 0s e 1s, que é inicializado como todos 0s e se torna uma solução viável da primal apenas na última iteração; e um vetor \mathbf{x} , não negativo, inicializado com 0s, e que é sempre viável para a dual do problema.
- (b) A cada iteração do algoritmo, você deve print o vetor \mathbf{y} em uma linha, o vetor \mathbf{x} na linha seguinte, e pular duas linhas. Arredonde no print o vetor \mathbf{x} para três casa decimais.
- (c) Após a última iteração, imprima também o custo da cobertura encontrada.

(ii) Ford-Fulkerson

- (a) A cada iteração, print (1) vetor 0s e 1s indicando qual caminho está sendo considerado, pule linha (2) vetor dos novos fluxos obtidos após a iteração, pule linha (3) cópia do vetor \mathbf{c} original, pule duas linhas. Todos esses vetores tem dimensão igual ao número de arcos.
- (b) Após a última iteração, print o valor máximo do fluxo encontrado, e pule linha.
- (c) Após o valor do fluxo máximo, print um vetor de 0s e 1s que indique um st -corte mínimo do grafo.
- (d) Todos números são inteiros, e portanto devem ser apresentados sem casas decimais.