Instruções para o TP2

Objetivo: implementar o algoritmo para encontrar emparelhamento perfeito de custo mínimo em grafo bipartido.

- Entrada: inteiro n indicando quantos vértices há em cada classe A e B do grafo bipartido, e inteiro m, indicando quantas arestas há no grafo. Matriz de incidência $N[i, j]$, tamanho $2n \times m$, do grafo bipartido, e vetor c de m entradas com os custos (inteiros positivos) das arestas, na ordem em que aparecem na matriz de incidência.

- Saída: caso não haja emparelhamento perfeito, retorne -1, seguido de um vetor de 0s e 1s de n entradas, em que entradas iguais a 1 indicam o conjunto $S \subseteq A$ que possui mais vértices do que sua vizinhança em B, seguido do vetor de 0s e 1s que indica a sua vizinhança. Caso haja emparelhamento perfeito, retorne o valor do emparelhamento de custo mínimo, seguido de vetor x com m entradas, todas 0s e 1s, onde 1 indica que a aresta pertence ao emparelhamento, e seguido de um certificado de que é um emparelhamento perfeito, ou seja, um vetor y que é viável para a dual da relaxação linear e que possui mesmo valor objetivo.

Instruções:

- Você basicamente precisa implementar o algoritmo descrito na seção 7.6. Ele parece bem simples à primeira vista, mas implícito neste algoritmo está a necessidade de implementar algo para testar se grafos bipartidos possuem emparelhamento perfeito ou não.

- Indique na sua entrega qual das opções abaixo você utilizou.

- Não é permitido copiar da internet qualquer algoritmo. Se você tiver dúvidas, recomendo que você assista vídeos explicando ou tire dúvidas com o professor ou o monitor, ao invés de pesquisar o código de um algoritmo. Será difícil "desvê-lo" porque são algoritmos com códigos relativamente curtos.

- Caso você já tenha implementado algum desses algoritmos no passado, pode utilizar sua versão, desde que, naturalmente, ela não tenha sido copiada de algum lugar... (passaremos comparador de códigos, então coisas vindo do github de alguém ou do geek4geeks ou similares podem dar match.. han han.. matchings.. haha). Indique na sua entrega se você implementou do zero ou se copiou de uma versão sua do passado.

- Você tem 3 opções:
- a ideal: implementar diretamente um algoritmo que encontra emparelhamentos perfeitos em grafos bipartidos. A ideia é a seguinte.
  - Comece com um emparelhamento M (no início, vazio) e olhe para um vértice não emparelhado em A.
  - Se ele possui vizinho não emparelhado em B, adicione esta aresta no emparelhamento e repita.
  - Agora você já tem um M maior, e assuma então que tem vértice v não emparelhado em A mas todos os seus vizinhos em B estão emparelhados por M com outros vértices de A.
  - Daí faça um BFS a partir de v, mas de modo que de um vértice de A para os de B, você considera todos os vizinhos possíveis, mas de B para A você só considera arestas do emparelhamento M.
  - Olha pras folhas da árvore após o BFS. Se alguma delas, digamos, u, está em B, então u não está emparelhado com ninguém de A (senão não seria folha), e você então olha pro caminho P entre v e u na árvore do BFS.
  - Daí você inverte as arestas de M que estão em P com as que não estão. O resultado será um emparelhamento maior (por que?). Daí você repete, etc.
  - Imagina agora que tem um vértice $v \in A$ que depois do BFS todas as folhas estão em A. Olhe para esta árvore inteira. Os vértices que estão em A tem todos os seus vizinhos (do grafo) nesta árvore, mas a árvore tem mais vértices em A do que em B....
- a 2a colocada: construir uma *st*-rede auxiliar, usar a correspondência entre fluxo máximo e emparelhamento máximo, e calcular o fluxo máximo usando Ford-Fulkerson. Note também que a saída de Ford-Fulkerson te dá um corte mínimo, e com ele você consegue um certificado que não há emparelhamento perfeito se for este o caso (note que esta opção não é muito diferente da anterior...)
- a 3a colocada (seu tp valerá no máximo 8 pontos): se o seu simplex do TP1 tá jóia, escreve a PI que calcula emparelhamento de tamanho máximo, e considera a relaxação linear dela. Como o grafo é bipartido, a saída do simplex será inteira. Se tiver n arestas, show. Se não tiver, o certificado de ótimo indica uma cobertura com menos do que n vértices.

O complemento dela no lado A é um conjunto S que tem todos os vizinhos na parte da cobertura que tá em B, mas $|S|$ será maior que sua vizinhança.

Entrada e Saída

Descrição da Entrada:

- A primeira linha da entrada irá conter dois valores inteiros, n e m, de forma que $1 < n \leq 20$ e $1 \leq m \leq 100$.
- Em seguida, as próximas $2n$ linhas com m colunas cada representam uma matriz de incidência Q.
- Cada linha representa um vértice v e cada coluna representa uma aresta a. A matriz Q é constituída de valores 0 e 1, ou seja, $Q_{u,v} \in \{0, 1\}$. Cada coluna possui apenas dois valores iguais a 1 indicando que a aresta liga esses dois o vértice.
- A última linha da entrada contém m valores com o peso de cada aresta na ordem que ela aparece na na matriz de incidência.

Descrição da Saída

Caso exista emparelhamento perfeito:

- A primeira linha deve conter o valor do emparelhamento perfeito de custo mínimo.
- A segunda linha deve conter m valores de 0 ou 1 que indicam se a aresta pertence ao emparelhamento.
- A terceira linha deve conter um certificado de que o emparelhamento é perfeito, ou seja, um vetor de tamanho $2n$ que é viável para a dual da relaxação linear e que possui mesmo valor objetivo.

Caso não exista um emparelhamento perfeito, a saída esperada serão três linhas:

- A primeira contém um -1
- A segunda deve conter um vetor de tamanho n que representa um subconjunto dos primeiros n vértices do grafo e que possua uma quantidade insuficiente de vizinhos para fazer o emparelhamento;
- A última linha deve conter o vetor de tamanho n que representa os vizinhos do subconjunto anterior.

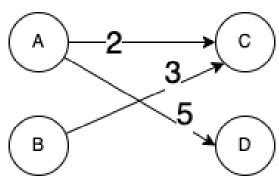
Exemplo 1

- Entrada

```
2 3
1 1 0
0 0 1
1 0 1
0 1 0
2 3 5
```

- Saída

```
8
0 1 1
2 5 0 1
```



Exemplo 2

- Entrada

```
2 2
1 0
0 1
1 1
0 0
2 3
```

- Saída

```
-1
1 1
1 0
```