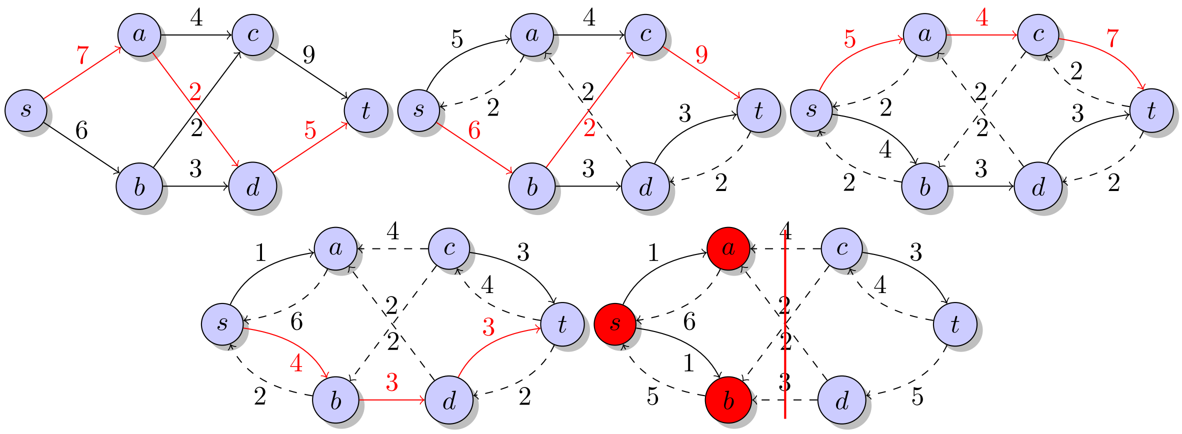
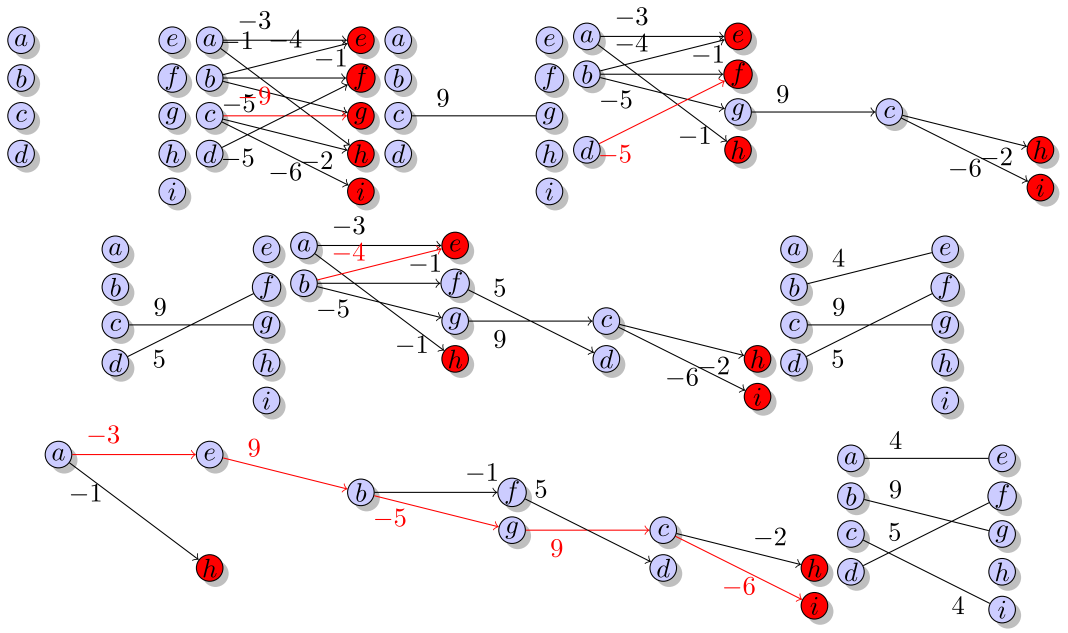
Universidade Federal do Rio Grande do Sul
Instituto de Informática
Departamento de Informática Teórica
CMP155 – Algoritmos e complexidade
2012/1
Prof. Marcus Ritt

Lista de soluções 4

Exercício 1
Mostraremos a sequencia de grafos residuais (com arcos backward pontilhados) e caminhos aumentantes (em vermelho) escolhidos durante a execução.



Exercício 2
Mostraremos a sequencia de emparelhamentos e árvores Húngaras com os vértices livres e o caminho mais curto em vermelho.



Exercício 3
O problema pode ser resolvido usando fluxo em redes. Forma um grafo com vértices V e uma cópia V' dos vértices em V, conectando todo vértice $v \in V$ com todo vértice $v' \in V$ tal que o vértice correspondente em V é um vizinho, com capacidade 1. Conecta um vértice fonte s com todos vértices $v \in V$ com capacidade c e cada vértice $v' \in V$ com um vértice destino t com capacidade l. Depois determina o fluxo máximo f nesse grafo. Caso $f(s) = c|V|$ existe uma solução do problema. Provaremos a corretude do algoritmo.

Proposição 0.1
O algoritmo acima termina com fluxo de valor $c|V|$ se e somente se o problema possui uma solução.

Prova. Dado uma solução do problema, podemos mandar um fluxo de c para cada vértice em V, que repassa um fluxo 1 para os seus c vizinhos escolhidos. Como nenhum vértice recebe mais que fluxo l, cada vértice em V' consegue repassar todo fluxo recebido para t, i.e., o valor do fluxo máximo e $c|V|$, já que $c|V|$ é um limite superior para o fluxo (dado pelo corte $(\{s\}, V \setminus \{s\})$.
Conversamente, supõe que o fluxo máximo possui valor $c|V|$. Pelo algoritmo de Ford-Fulkerson sabemos que existe um fluxo integral com o mesmo valor. Como todo vértice em V recebe fluxo c, mas pode passar somente fluxo 0 ou 1 para um vértice em V', ele tem que mandar um fluxo 1 para exatamente c vizinhos, que podemos escolher para formar a solução do problema. Nenhum dos vizinhos pode ser selecionado mais que l vezes, porque a capacidade sainte de cada vértice em V' é l. ■

Exercício 4
O problema pode ser resolvido calculando um emparelhamento máximo num grafo bipartido em tempo $O(m\sqrt{n})$ usando o algoritmo de Hopcroft-Karp. Forma uma grafo bipartido com vértices V é uma cópia V' dos vértices em V, conectando todo vértice $v \in V$ com todo vértice $v' \in V$ tal que o vértice correspondente em V é um vizinho e determina o emparelhamento máximo neste grafo.

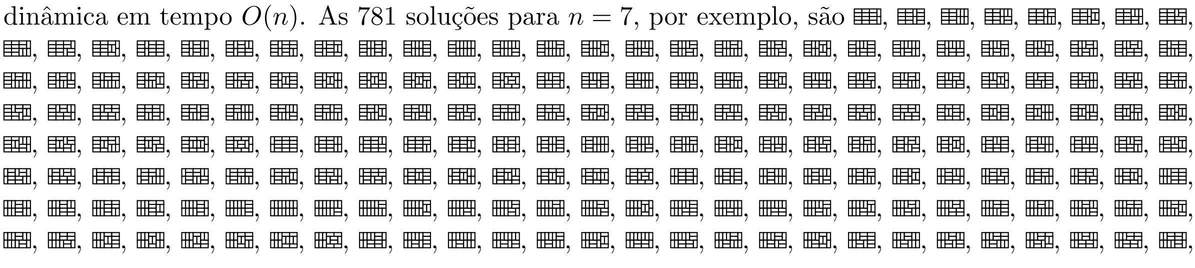
Proposição 0.2
O grafo possui uma cobertura por ciclos sse o emparelhamento máximo é perfeito.

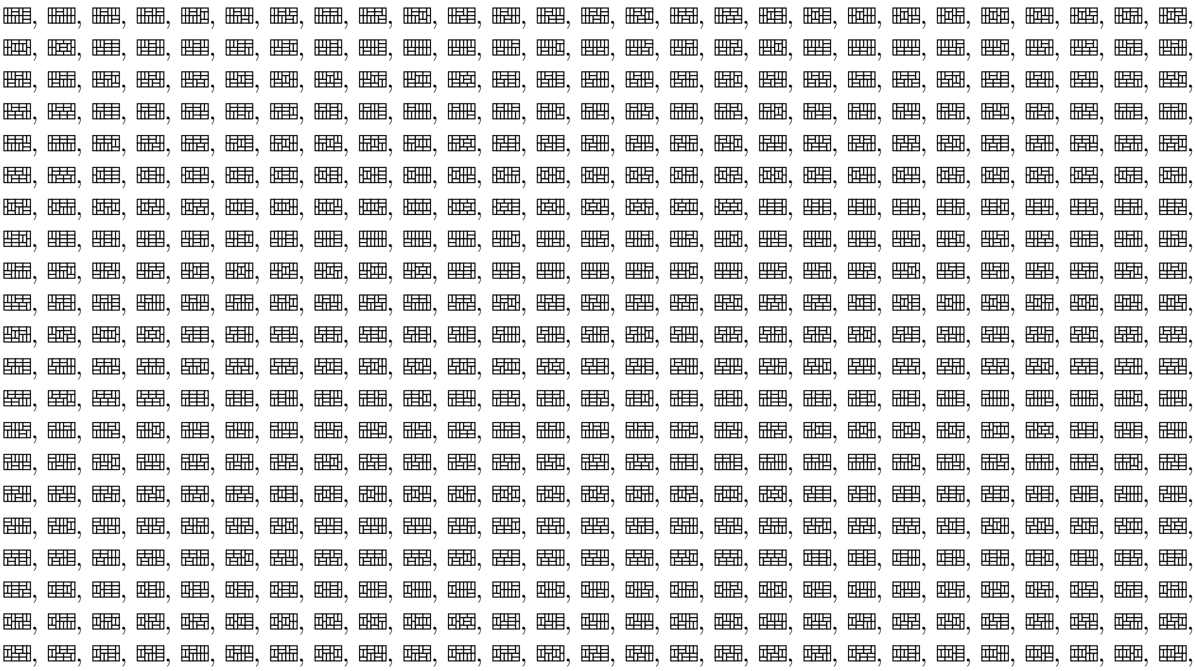
Prova. Caso o grafo possui um cobertura por ciclos, para cada arco (u, v) emparelha vértices $u \in V$ e $v' \in V'$. Como cada vértice possui exatamente um arco sainte e entrante isso forma um emparelhamento perfeito.
Caso temos um emparelhamento perfeito no grafo definido acima podemos extrair ciclos do emparelhamento como segue. Escolhe um vértice que não faz parte de nenhum ciclo ainda. Como o emparelhamento é perfeito ele contém uma aresta (u, v'). Insere o arco (u, v) no ciclo atual e continua com v. Esse processo tem que terminar fechando um ciclo. Repete com outro vértice ainda livre até todos vértices foram visitados. Nenhum vértice faz parte de dois ciclos escolhidos dessa maneira, senão algum vértice teria que ter dois predecessores diferentes em ciclos diferentes, e logo teria que ser emparelhado com dois vértices diferentes. ■

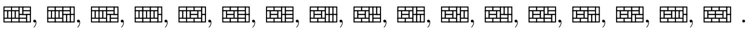
Exercício 5
A primeira coluna pode ser coberta por , , , , ou . A segunda opção gera um retângulo cortado que pode continuar com ou . Observe que a primeira continuação fecha uma coluna completamente, enquanto a segunda gera um retângulo cortado bem como a quarta opção inicial. Esse retângulo pode continuar com ou . Finalmente a terceira opção inicial gera um retângulo cortado que possui continuações ou . Logo podemos definir as seguintes recorrências para o número de coberturas T_n de um retângulo com n colunas, o número de coberturas U_n de um retângulo cortado , o número de coberturas T_n de um retângulo cortado e o número W_n para um retângulo cortado

$$\begin{aligned}
T_n &= T_{n-2} + U_{n-1} + W_{n-1} + V_{n-1} + T_{n-1}\\
U_n &= T_{n-1} + V_{n-1}\\
V_n &= T_{n-1} + U_{n-1}\\
W_n &= T_{n-1} + W_{n-2}.
\end{aligned}$$

com base $T_1 = U_1 = V_1 = W_1 = 1$ e $W_0 = 0$. Tais valores podem ser calculadas usando programação dinâmica em tempo $O(n)$. As 781 soluções para $n = 7$, por exemplo, são





 . A resposta para $n = 10$ é 18061.