Sobre o "Método Húngaro"

Marcus Ritt

Março 2023

Outline

Cuidado

- Tem diversos algoritmos que são encontrados pelo nome de "Método Húngaro"
- O nosso método de resolver o problema segue o livro de Schrijver (Combinatorial Optimization, 2004, cáp. 17, "Weighted bipartite matching and the assignment problem")
- Em particular tem outros métodos bem diferentes, cujo princípio é de algoritmos primais-duais
  - Nos não discutimos este princípio em aula, então não tem como entender estes métodos
- O objetivo do trabalho não é implementar qualquer algoritmo encontrado pelo nome "Método Húngaro", mas o algoritmo discutido em aula, e em particular a variante que busca caminhos aumentantes mais curtos usando a transformação de Johnson e o algoritmo de Dijkstra

O método de Johnson

- Para lembrar: caso um grafo não tem ciclos negativos, é possível definir um *potencial* p_v nos vértices $v \in V$ tal que as distâncias transformadas $d'_{uv} = d_{uv} - (p_v - p_u) \geq 0$, i.e.~são não-negativas.
- Uma vez na posse de um potencial desses, então, podemos simplesmente rodar o algoritmo de Dijkstra (como implementado no 1o trabalho)
- O problema com isso: descobrir o potencial ab initio custa tempo $O(nm)$; isso não faz sentido, porque o algoritmo de Bellmann-Ford resolve o problema dos caminhos mais curtos diretamente em tempo $O(nm)$. (Isso é somente útil para resolver o problema de encontrar as caminhos mais curtos entre todos pares de vértices).
- Logo o problema é: manter um potencial durante a execução do algoritmo de forma mais eficiente

Manter um potencial de forma eficiente I

- Isso funciona como segue
- Inicialmente:
  - Para todo vértice $v \in S$: define $p_v := 0$, onde $W := \max_{e \in E} w_e$.
  - Para todo vértice $v \in T$: define $p_v := -W$.
  - Isso satisfaz a condição, porque a orientação de todos arcos e é de S para T e logo tem peso $-w_e$ e ainda $d'_{uv} = d_{uv} - (p_v - p_u) = d_{uv} + W \geq 0$.
- Em cada iteração, na posse de um potencial correto:
  - Roda o algoritmo de Dijsktra iniciando com os vértices livres S_0 em S.
  - Isso determina dist'(S_0, v) para todos vértices v (nas distâncias modificadas, por isso dist').
  - Encontra a caminho mais curto que termina num vértice livre em T.

Manter um potencial de forma eficiente II

- Depois atualiza atualiza o potencial como segue
  - Simplesmente define $p_v := \text{dist}(S_0, v)$ usando o caminho mais curto encontrado
  - Nota que isso são os caminhos mais curtos mas com distâncias usando os pesos originais (por isso $\text{dist}(S_0, v)$).
  - Para atualizar
    - Temos $\text{dist'}(S_0, v) = \text{dist}(S_0, v) - (p_v - p_{s_0}) = \text{dist}(S_0, v) - p_v$
    - Logo: o novo potencial simplesmente é $\text{dist}(S_0, v) = p_v + \text{dist'}(S_0, v)$.
  - Isso nunca gera um problema porque
    - O conjunto de vértices livres em S diminui em cada iteração, $S_0' \subseteq S_0$.
    - O conjunto de vértices livres em T diminui em cada iteração, $T_0' \subseteq T_0$.