Análise e Síntese de Algoritmos

Algoritmos de Aproximação [CLRS, Cap. 35]

2011/2012

Contexto

- Revisão [CLRS, Cap.1-13]
  - Fundamentos; notação; exemplos
- Algoritmos em Grafos [CLRS, Cap.21-26]
  - Algoritmos elementares
  - Árvores abrangentes
  - Caminhos mais curtos
  - Fluxos máximos
- Programação Linear [CLRS, Cap.29]
  - Algoritmos e modelação de problemas com restrições lineares
- Técnicas de Síntese de Algoritmos [CLRS, Cap.15-16]
  - Programação dinâmica
  - Algoritmos greedy
- Tópicos Adicionais [CLRS, Cap.32-35]
  - Emparelhamento de Cadeias de Caracteres
  - Complexidade Computacional
  - Algoritmos de Aproximação

Resumo

Motivação

Motivação

- Problemas NP-Completos
  - Qualquer algoritmo existente demora no pior caso tempo exponencial no tamanho da instância
  - Suporta conjectura $P \neq NP$
- Problemas de decisão
  - Formulação original requer resposta sim/não
- Problemas de optimização
  - Formulação de decisão provada NP-completa
  - Encontrar solução óptima é computacionalmente difícil
    - Calcular (em tempo polinomial) solução aproximada !
    - Mas estabelecer garantias quanto ao valor da solução calculada

Motivação

Algoritmos de Aproximação

- Algoritmos, com complexidade polinomial, que calculam soluções aproximadas para problemas de optimização NP-difíceis
  - i.e. problemas de optimização cuja formulação como problema de decisão é NP-completa
  - Existem garantias quanto ao máximo afastamento do valor calculado face à solução óptima

Definições

Definições

- Limite da razão $\rho(n)$
  - Algoritmo de aproximação para um problema de optimização
  - Para qualquer instância de tamanho n
    - Custo da solução calculada, C
    - Custo da solução óptima, C^*
  - $max(C/C^*, C^*/C) \leq \rho(n)$
    - maximização: $0 \leq C \leq C^*$
    - minimização: $0 \leq C^* \leq C$
    - $\rho(n) \geq 1$

Problema da Cobertura de Vértices

Problema Cobertura de Vértices (VC)

- Definição:
  - $G = (V, E)$, grafo não dirigido
  - Cobertura de vértices (VC): conjunto de vértices V' tal que qualquer arco em G é incidente em pelo menos um dos vértices de V'
  - O problema VC_{opt} consiste em calcular uma cobertura de vértices com o número mínimo de vértices

Problema da Cobertura de Vértices

Problema Cobertura de Vértices (VC)

Algoritmo de Aproximação

VC-Approx(G)

1 $C = \emptyset$
2 $E' = E[G]$
3 **while** $E' \neq \emptyset$
4 **do** Seja (u, v) arco de E'
5 $C = C \cup \{u, v\}$
6 Remover de E' qualquer arco incidente em u ou v
7 **return** C

Problema da Cobertura de Vértices

Problema Cobertura de Vértices (VC)

- Teorema:
  - VC-Approx é um algoritmo de aproximação com limite da razão 2 para o problema VC_{opt}
    - $\max(C/C^*, C^*/C) = C/C^* \leq 2$

Problema da Cobertura de Vértices

Problema Cobertura de Vértices (VC)

- Prova:
  - A: conjunto de arcos seleccionados pelo algoritmo
  - Conjunto C é cobertura de vértices
    - Ciclo iterado até todos os arcos de E estarem cobertos
  - Por inspecção, nenhum par de arcos em A tem vértices comuns
  - Também por inspecção, $|C| = 2|A|$
  - Cobertura dos arcos em A requer pelo menos um vértice incidente em cada um dos arcos de A
    - Qualquer cobertura de vértices tem que cobrir arcos de A
    - Válido também para C^*
    - $|A| \leq |C^*|$
  - Conclusão: $|C| = 2|A| \leq 2|C^*|$

Problema da Mochila

Problema da Mochila

- Definição:
  - Dados n objectos com pesos $w_1, w_2, \ldots, w_n$ e valores $v_1, v_2, \ldots, v_n$, e uma mochila que pode transportar peso máximo W, o problema $KNAPSACK_{opt}$ consiste em identificar os objectos a transportar, que não excedem o peso máximo W, e que maximizam o valor dos objectos transportados

Problema da Mochila

Problema da Mochila

Algoritmo Greedy

KNAP-Greedy(w, v, W)

```
1  Ordenar w e v tal que v[i]/w[i] ≥ v[j]/w[j] para 1 ≤ i ≤ j ≤ n
2  peso = 0
3  valor = 0
4  for i = 1 to n
5        do if peso + w[i] ≤ W
6              then valor += v[i]
7                   peso += w[i]
8  return valor
```

Problema: Solução Greedy pode ficar arbitrariamente longe do valor óptimo do problema.

Problema da Mochila

Problema da Mochila

Algoritmo de Aproximação

KNAP-Approx(w, v, W)

1 $maior = \max\{v[i] : 1 \leq i \leq n\}$ ▷ Admite-se $w[i] \leq W, 1 \leq i \leq n$
2 **return** $\max\{maior, \text{KNAP-Greedy}(w, v, W)\}$

Problema da Mochila

Problema da Mochila

- Teorema:
  - O algoritmo KNAP-Approx é um algoritmo de aproximação com limite da razão 2 para o problema $KNAPSACK_{opt}$
- Prova:
  - C^*: solução óptima; C: solução retornada pelo algoritmo
  - Escolher menor k tal que, $\sum_{i=1}^{k} w[i] \geq W$
  - KNAP-Greedy(w, v, W) verifica, KNAP-Greedy$(w, v, W) \geq \sum_{i=1}^{k-1} v[i]$
  - E, *maior* $\geq v_k$ (por definição de maior)

Problema da Mochila

Problema da Mochila

- Notando que $max(x, y) \geq (x + y)/2$, obtemos:

$$\begin{aligned} C &= \max\{maior, \text{KNAP-Greedy}(w, v, W)\} \\ &\geq (maior + \text{KNAP-Greedy}(w, v, W))/2 \\ &\geq (v_k + \sum_{i=1}^{k-1} v[i])/2 \\ &= (\sum_{i=1}^{k} v[i])/2 = C'/2 \\ &\geq C^*/2 \end{aligned}$$

- Definição de W' e C':
  - $W' = \sum_{i=1}^{k} w[i] \qquad C' = \sum_{i=1}^{k} v[i]$
- Notar que $W' \geq W$ e $v_1/w_1 \geq v_2/w_2 \geq \ldots \geq v_k/w_k$
- Pelo que $C' \geq C^*$, dado que C' é a solução óptima para W'

Problema do Caixeiro Viajante

Problema do Caixeiro Viajante

- Definição:
  - $G = (V, E)$, grafo completo não dirigido, com função de pesos w
  - Circuito: caminho que visita todos os vértices de G apenas uma única vez e que termina no vértice de onde começa
    - Peso de um circuito: peso do caminho respectivo
  - Problema do Caixeiro Viajante (TSP):
    - Encontrar o circuito de menor peso em G
  - Desigualdade triangular:
    - $w(t, v) \leq w(t, u) + w(u, v)$ para quaisquer arcos $(t, v), (t, u)$ e (u, v) de G

Problema do Caixeiro Viajante

Problema do Caixeiro Viajante

Algoritmo de Aproximação

TSP-Approx(G, w)

1 Seleccionar qualquer $r \in V$ para vértice raiz
2 Construir MST T a partir de r
3 L = lista de vértices resultantes de visita de T em pré-ordem
4 Definir circuito H que visita vértices de G pela ordem L
5 **return** H

Pré-ordem: Visita de árvore que lista a raiz antes dos vértices de cada sub-árvore

Problema do Caixeiro Viajante

Problema do Caixeiro Viajante

- Teorema:
  - O algoritmo TSP-Approx é um algoritmo de aproximação com limite da razão 2 para instâncias do problema TSP que verificam a desigualdade triangular
- Prova:
  - H^*: circuito com custo mínimo
    - Provar: $C(H) \leq 2C(H^*)$
  - T: MST de G
  - W: travessia completa de T
    - cada vértice u identificado se visitado pela primeira vez ou se travessia retorna a u

Problema do Caixeiro Viajante

Problema do Caixeiro Viajante

- $C(T) \leq C(H^*)$
  - T é MST de G e qualquer árvore abrangente pode ser obtida a partir de circuito por remoção de 1 arco
- $C(W) = 2C(T)$
  - W atravessa cada arco 2 vezes
- Pelo que,
  - $C(W) \leq 2C(H^*)$
- Ordem dos vértices visitados em W:
  - $\ldots, t, v, u, \ldots, u, v, t, \ldots$
- Desigualdade triangular assegura que podemos apagar um vértice que o custo total não aumenta
  - E.g.: com $\ldots, t, v, u, \ldots, u, t, \ldots$ custo total não aumenta

Problema do Caixeiro Viajante

Problema do Caixeiro Viajante

- Repetir processo de remoção de vértices até cada vértice ser visitado apenas 1 vez
  - Obtém-se ordem dos vértices após visita em pré-ordem
- Circuito resultante, H, visita todos os vértices de G e verifica:
  - $C(H) \leq C(W) \leq 2C(H^*)$

Problema da Cobertura de Conjuntos

Problema da Cobertura de Conjuntos

- Definição:
  - Uma instância (X, F) do problema de cobertura de conjuntos consiste de:
    - Conjunto finito X
    - Família de subconjuntos de X, F
    - Tal que, $X = \cup_{S \in F} S$
  - O problema da cobertura de conjuntos consiste em identificar um subconjunto C de F de menor tamanho e tal que $X = \cup_{S \in C} S$

Problema da Cobertura de Conjuntos

Problema da Cobertura de Conjuntos

Algoritmo de Aproximação

SC-Approx(X, F)

1 $U = X$
2 $C = \emptyset$
3 **while** $U \neq \emptyset$
4 **do** Seleccionar $S \in F$ que maximiza $|S \cap U|$
5 $U = U - S$
6 $C = C \cup \{S\}$
7 **return** C

Problema da Cobertura de Conjuntos

Problema da Cobertura de Conjuntos

- Teorema:
  - O algoritmo SC-Approx tem limite da razão $H(\max\{|S| : S \in F\})$, com $H(d) = 1 + 1/2 + \ldots + 1/d, H(0) = 0$
- Prova: (ver CLRS)
- Corolário:
  - SC-Approx tem limite da razão de ln |X| + 1
- Prova: (ver CLRS)