Algorithms and Complexity

6. Approximation Algorithms

Marc Hellmuth

University of Stockholm

Approximation Algorithms

NP complete problems

- no polynomial time solution known
- but many problems too important to ignore

Approximation Algorithms

NP complete problems

- no polynomial time solution known
- but many problems too important to ignore
- work-arounds:
  - use exponential time solution for small problem instances
  - identify special cases in which a polynomial time solution exists
  - approximate optimal solution

Exmpl: Vertex Cover

Two possible heuristics:

Greedy_VC1($G = (E, V)$)

```
C = ∅, E' = E
while E' ≠ ∅ do
    v = vertex of max degree in G' = (V, E')
    C = C ∪ {v}
    Remove all edges invident to v from E'
return C
```

Greedy_VC2($G = (E, V)$)

```
C = ∅, E' = E
while E' ≠ ∅ do
    e = {u, w} some edge in E'
    C = C ∪ {u, w}
    Remove all edges incident to u
and w from E'
return C
```

Which one is "better", that is, in general closer to an optimal solution?

As we shall see the "smarter" method Greedy_VC1 does not yield solutions that are, in general, close to the optimal one, while Greedy_VC2 always yields a vertex cover C that is never larger than two times the optimal solution.

WHITEBOARD

ρ-Approximation Algorithm

Definition 1

- Π: Optimization problem
- $I \in \Pi$ an instance of Π
- $A(I)$: return value of algorithm A applied on instance I
- $\textsc{Opt}(I)$: optimal value of instance I

Definition 2

Algorithm A has approximation-ratio $\rho \in \mathbb{R}_{\geq 1}$ if for all instances $I \in \Pi$ it holds that

$$\frac{1}{\rho}\textsc{Opt}(I) \leq A(I) \leq \rho\textsc{Opt}(I)$$

Such an algorithm is called ρ-approximation algorithm

For minimzation problems we only need to show: $\frac{A(I)}{\textsc{Opt}(I)} \leq \rho$

For maximization problems we only need to show: $\frac{\textsc{Opt}(I)}{A(I)} \leq \rho$

Exmpl: Traveling-salesperson problem (TSP)

Let $G = (V, E)$ be a complete undirected graph in which each edge $e \in E$ has weight $w(e) \geq 0$.
Find a Hamiltonian cycle ("tour", a cycle that visits every vertex exactly once) of G with minimum cost.

Definition 3 (Triangle inequality)

We say that the weight function w satisfies the triangle inequality if

$$w(\{a,c\}) + w(\{b,c\}) \geq w(\{a,b\})$$

for all vertices $a, b, c \in V$.

Definition 4

$\triangle$-TSP denotes the set of all instances of TSP for which the weights satisfy the triangle inequality.

Exercise: $\triangle$-TSP remains NP-complete

Approximating △-TSP

APPROX-TSP

construct a minimum spanning tree T for G
let H = cycle "visiting the vertices preorder walk" in T
return the Hamiltonian cycle H

$T = (V, E)$ = tree
v.visited = false $\forall v \in V$
$x \in V$

PREORDER-WALK(x)

if x.visited = false **then**
    print x
    x.visited = true
    for all $v \in N(x)$ in T **do**
        PREORDER-WALK(v)

WHITEBOARD

Theorem 6.1

APPROX-TSP *is a polynomial-time 2-approximation algorithm for* $\triangle$*-TSP.*

proof: WHITEBOARD

Theorem 6.2

If $P \neq NP$*, then there is no polynomial-time* ρ*-approximation algorithm for TSP for all* $\rho \geq 1$

proof: WHITEBOARD

Approximation scheme

Definition 5 (Approximation scheme)

An approximation scheme is an approximation algorithm that takes as input instances of an optimization problem Π and an $\varepsilon > 0$, such that for any $\varepsilon > 0$, the scheme is a $(1+\varepsilon)$-approximation algorithm for Π.

Definition 6 (Polynomial-time approximation scheme)

An polynomial-time approximation scheme (PTAS) is an approximation scheme that runs in polynomial time in n for any $\varepsilon > 0$.

Example 7

$O(n^{\exp(1/\varepsilon)})$

Approximation scheme

Definition 8 (Fully polynomial-time approximation scheme)

A fully polynomial-time approximation scheme (FPTAS) is an approximation scheme whose running time is polynomial in both the size of the input I and $1/\varepsilon$.

Example 9

$O(n^5(1/\varepsilon)^2)$

Remark

In a fully polynomial-time approximation scheme a decrease of ε by a constant factor leads to an increase in the running time by at most a(nother) constant factor.

Exmpl: Simple Knapsack

SIMPLE_KNAPSACK (SK)

Input: Set of item $W = \{1, \ldots, n\}$, Capacity C, value(=weight) of item $w_1 \ldots, w_n \in \mathbb{N}$, integer B

Question: $\exists M \subset W$ such that $B \leq \sum_{i \in M} w_i \leq C$

Exercise: SK is NP-complete

GREEDY_SK($W = \{1, \ldots, n\}$, weights $w_1 \geq w_2 \geq \cdots \geq w_n$, C)

$M = \emptyset$, value$= 0$
for $i = 1 \ldots n$ **do**
    if value $+ w_i \leq C$ **then**
        $M = M \cup \{i\}$
        value $=$ value $+ w_i$
return M, value

Theorem 6.3

GREEDY_SK *is a polynomial-time 2-approximation algorithm for SK.*

Proof: WHITEBOARD

Exmpl: Simple Knapsack

SK-SCHEME($W = \{1, \dots, n\}$, weights $w_1 \geq w_2 \geq \dots \geq w_n$, C, ε)

1: $k_\varepsilon = \lceil \frac{1}{\varepsilon} \rceil$, value$= 0$
2: **for** all subsets $M \subseteq W$ with $|M| \leq k_\varepsilon$ and $\sum_{i \in M} w_i \leq C$ **do**
3: Extend M via GREEDY_SK to M^*, that is, add to $M = \{i_1, \dots, i_p\}$ greedily $i_{p+1}, \dots, n$
4: **return** the M^* for which $\sum_{i \in M^*} w_i$ is maximum

Theorem 6.3

SK-SCHEME *is a PTAS for SK.*

Proof: WHITEBOARD