Inapproximability

Design and Analysis of Algorithms
Andrei Bulatov

Center Selection and Friends

Metric Center Selection

Instance:

A set V of n sites, distances satisfying the triangle inequality, k, the number of centers

Objective:

Find a set $S \subseteq V$ such that the maximal (over all sites) distance from a site to a closest center is as small as possible

Dominating Set

Instance:

A graph $G = (V,E)$.

Objective:

Find a smallest dominating set in G, i.e. a set adjacent to all nodes in G

Center Selection: Hardness of Approximation

Theorem

Unless P = NP, there is no ρ-approximation algorithm for Metric k-Center problem for any $\rho < 2$. (k is considered a part of the input.)

Proof

We show how we could use a $(2 - \varepsilon)$-approximation algorithm for k-Center to solve DOMINATING-SET in poly-time.

Let G = (V, E), k be an instance of DOMINATING-SET

Construct instance G' of k-center with sites V and distances

$d(u, v) = 1$ if $(u, v) \in E$

$d(u, v) = 2$ if $(u, v) \notin E$

Note that G' satisfies the triangle inequality.

Center Selection: Hardness of Approximation

Proof (cntd)

Claim:

G has dominating set of size k iff there exists k centers C^* with $r(C^*) = 1$.

Thus, if G has a dominating set of size k, a $(2 - \varepsilon)$-approximation algorithm on G' must find a solution C^* with $r(C^*) = 1$ since it cannot use any edge of distance 2.

QED

TSP

Theorem

Unless P = NP, TSP is not approximable

Proof

Suppose for contradiction that there is an $(1+\varepsilon)$-approximating algorithm for TSP; that is, for any collection of cities and distances between them, the algorithm finds a tour of length l such that

$$\frac{l - \text{OPT}}{\text{OPT}} \le \varepsilon$$

We use this algorithm to solve Hamiltonian Cycle in polynomial time

TSP

For any graph G = (V,E), construct an instance of TSP as follows:

- Let the set of cities be V
- Let the distance between a pair of cities v_1, v_2 be

$$d(v_1, v_2) = \begin{cases} 1 & \text{if } (v_1, v_2) \in E \\ 2(1+\varepsilon)|V| & \text{otherwise} \end{cases}$$

- If G has a Hamilton Cycle, then it has a tour of length |V|
- Otherwise the minimal tour is at least $2(1+\varepsilon)|V|$

Hence the (1+ε)-approximating algorithm would find a tour of length l such that

$$\frac{l}{\text{OPT}} - 1 \le \varepsilon \qquad \Rightarrow \qquad l \le (1+\varepsilon) \cdot \text{OPT}$$

More Inapproximability

Maximum Independent Set

Instance:

A graph G = (V,E).

Objective:

Find a largest set $M \subseteq N$ such that no two vertices from M are connected

Maximum Clique

Instance:

A graph $G = (V,E)$.

Objective:

Find a largest clique in G

Independent Set vs. Clique

Observation

For a graph G with n vertices, the following conditions are equivalent

- G has a vertex cover of size k
- G has an independent set of size n – k
- $\overline{G}$ has a clique of size n – k

Theorem

Unless P = NP, Max Independent Set and Max Clique are not approximable

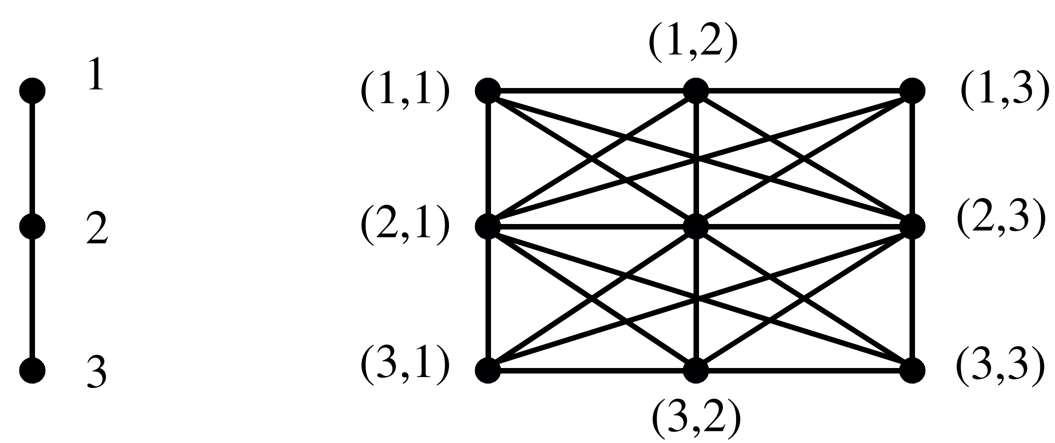
Proof

We prove a weaker result:

If there is an (1-ε)-approximating algorithm for Max Independent Set then there is a FPAS for this problem

For a graph G = (V,E), the square of G is the graph G^2 such that

- its vertex set is $V \times V = \{(u,v) \mid u,v \in V\}$
- $\{(u,u'),(v,v')\}$ is an edge if and only if $\{u,v\} \in E$ or $u = v$ and $\{u',v'\} \in E$



Independent Set: Hardness of Approximation

Lemma

A graph G has an independent set of size k if and only if G^2 has a independent set of size k^2

Proof

If I is an independent set of G then $\{(u,v) \mid u,v \in I\}$ is an independent set of G^2

Conversely, if I^2 is an independent set of G^2 with k^2 vertices, then

- $I = \{u \mid (u,v) \in I^2$ for some $v\}$ is an independent set of G
- $I_u = \{v \mid (u,v) \in I^2\}$ is an independent set of G

Proof (cntd)

Suppose that a (1-ε)-approximating algorithm exists, working in $O(n^l)$ time

Let G be a graph with n vertices, and let a maximal independent set of G has size k

Applying the algorithm to G^2 we obtain an independent set of G^2 of size $(1-\varepsilon)k^2$ in a time $O(n^{2l})$

By Lemma, we can get an independent set of G of size $\sqrt{1-\varepsilon}\cdot k$

Therefore, we have an $\sqrt{1-\varepsilon}$ -approximating algorithm

Repeating this process m times, we obtain a $\sqrt[2^m]{1-\varepsilon}$-approximation algorithm working in $O(n^{2^m l})$ time

Proof (cntd)

Given ε' we need m such that

$$(1-\sqrt[2^m]{1-\varepsilon}) < \varepsilon'$$

$$\sqrt[2^m]{1-\varepsilon} > 1-\varepsilon'$$

$$\frac{\log(1-\varepsilon)}{2^m} > \log(1-\varepsilon')$$

$$\frac{1}{2^m} < \frac{\log(1-\varepsilon')}{\log(1-\varepsilon)}$$

$$m > \log\frac{\log(1-\varepsilon)}{\log(1-\varepsilon')}$$

Then our ε′-approximating algorithm works in a time $O\left(n^{l\frac{\log(1-\varepsilon)}{\log(1-\varepsilon')}} \right)$

FPTAS

Design and Analysis of Algorithms

Andrei Bulatov

Polynomial Time Approximation Scheme

- PTAS. An approximation algorithm for any constant relative error $1 \pm \varepsilon > 0$.
  - Load balancing. [Hochbaum-Shmoys 1987]
  - Euclidean TSP. [Arora 1996]

- Consequence. PTAS produces arbitrarily high quality solution, but trades off accuracy for time.

- FPTAS (Fully polynomial approximation scheme)
  if the algorithm is polynomial time in the size of the input and $1/\varepsilon$

Knapsack

The Knapsack Problem

Instance:

A set of n objects, each of which has a positive integer value v_i and a positive integer weight w_i. A weight limit W.

Objective:

Select objects so that their total weight does not exceed W, and they have maximal total value

Example: { 3, 4 } has value 40.

W = 11

Item	Value	Weight
1	1	1
2	6	2
3	18	5
4	22	6
5	28	7

Knapsack: Dynamic Programming II

OPT(i, v) is min weight subset of items 1, …, i of value exactly v.

Case 1: OPT does not select item i.

OPT selects best of 1, …, i – 1 that achieves exactly value v

Case 2: OPT selects item i.

consumes weight w_i, new value needed is $v - v_i$

OPT selects best of 1, …, i – 1 that achieves exactly value $v - v_i$

$$OPT(i,v) = \begin{cases} 0 & \text{if } v = 0 \\ \infty & \text{if } i = 0, v > 0 \\ OPT(i-1,v) & \text{if } v_i > v \\ \min\{OPT(i-1,v), w_i + OPT(i-1, v - v_i)\} & \text{otherwise} \end{cases}$$

$V^* \leq n\, v_{max}$

Running time. $O(nV^*) = O(n^2 v_{\max})$

V* = optimal value = maximum v such that OPT(n, v) ≤ W.

Not polynomial in input size!

Knapsack: FPTAS

Intuition for approximation algorithm.

- Round all values up to lie in smaller range.
- Run dynamic programming algorithm on rounded instance.
- Return optimal items in rounded instance.

Item	Value	Weight
1	1,734,221	1
2	6,656,342	2
3	18,810,013	5
4	22,217,800	6
5	28,343,199	7

W = 11

original instance

⇨

Item	Value	Weight
1	2	1
2	7	2
3	19	5
4	23	6
5	29	7

W = 11

rounded instance

Knapsack: FPTAS

Knapsack FPTAS. Round up all values:

- v_{max} = largest value in original instance
- ε = precision parameter
- θ = scaling factor = $\varepsilon\, v_{max} / n$

$$\bar{v}_i = \left\lceil \frac{v_i}{\theta} \right\rceil \theta, \; \hat{v}_i = \left\lceil \frac{v_i}{\theta} \right\rceil$$

Observation. Optimal solution to problems with $\bar{v}$ or $\hat{v}$ are equivalent.

Intuition. $\bar{v}$ close to v so optimal solution using $\bar{v}$ is nearly optimal; $\hat{v}$ small and integral so dynamic programming algorithm is fast.

Running time. $O(n^3 / \varepsilon)$.

– Dynamic program II running time is $O(n^2\, \hat{v}_{max})$, where

$$\hat{v}_{max} = \left\lceil \frac{v_{max}}{\theta} \right\rceil = \left\lceil \frac{n}{\varepsilon} \right\rceil$$

Knapsack: FPTAS

Knapsack FPTAS. Round up all values: $\bar{v}_i = \left\lceil \frac{v_i}{\theta} \right\rceil \theta$

Theorem

If S is the solution found by our algorithm and S* is any other feasible solution then $(1+\varepsilon)\sum_{i \in S} v_i \geq \sum_{i \in S^*} v_i$

Proof:

Let S* be any feasible solution satisfying weight constraint

Knapsack: FPTAS

$$\sum_{i \in S^*} v_i \le \sum_{i \in S^*} \overline{v}_i$$ always round up

$$\le \sum_{i \in S} \overline{v}_i$$ solve rounded instance optimally

$$\le \sum_{i \in S} (v_i + \theta)$$ never round up by more than θ

$$\le \sum_{i \in S} v_i + n\theta$$ $|S| \le n$

DP alg can take v_{max}

$$\le (1+\varepsilon) \sum_{i \in S} v_i$$ $n\,\theta = \varepsilon\, v_{max}$, $v_{max} \le \Sigma_{i \in S}\, v_i$