Polynomial-Time Approximation Scheme

DANG Vu Lam   GUILLOTEAU Quentin

Monday 1st April, 2019

Table of Contents

PTAS: Definition

PTAS: Definition

Definition (Minimization problem)

A PTAS is an algorithm A, which takes an instance of an optimization problem I and a parameter $\varepsilon > 0$ and produces, **in polynomial time** (in n for a fixed ε) a solution
$A(I) \leq (1 + \varepsilon) \times OPT(I)$

Definition (Maximization problem)

A PTAS is an algorithm A, which takes an instance of an optimization problem I and a parameter $\varepsilon > 0$ and produces, **in polynomial time** (in n for a fixed ε) a solution
$A(I) \geq (1 - \varepsilon) \times OPT(I)$

Where $OPT(I)$ is the optimal solution of the optimization problem for the instance I.

FPTAS and Asymptotic PTAS: Definition

Definition (FPTAS)

A FPTAS (Fully Polynomial-Time Approximation Scheme) is a PTAS that runs in polynimial time in n and in $\frac{1}{\varepsilon}$

Definition (Asymptotic PTAS)

An Asymptotic PTAS is an algorithm A, which takes an instance of an optimization problem I and a parameter $\varepsilon > 0$ and produces, **in polynomial time** (in n for a fixed ε) a solution
$A(I) \leq (1 + \varepsilon) \times OPT(I) + C(\varepsilon)$

Application to the Bin packing problem

The Bin packing problem

The Bin Packing Problem (BPP)

Setup

Given a set of bins $S_1, S_2 \ldots$ with the same size 1 and a set of n items with sizes $s_1, s_2, \ldots, s_n$ and $\forall i, 1 \leq i \leq n, s_i \in [0, 1]$.

Goal

Find

- the minimal (integer) number of bins B
- a B-partition $S_1 \bigcup \cdots \bigcup S_B$ of $\{1, \ldots, n\}$ such that: $\forall k \in \{1, \ldots, B\}, \sum_{i \in S_k} s_i \leq 1$

First Fit Algorithm

First Fit Algorithm

Algorithm

1. For each items s_i find the first bin that fit
2. If no bins fit, order new bin
3. Until no more items left

Remark

- 2 - approximation
- *How can we improve?*

Restricted instances

PTAS for BPP - Number of sizes is less than n

Theorem

For all instances of the BPP where the number of different item-sizes is K ($K < n$), there is a **polynomial algorithm** for solving these instances.

Proof

We can represent a bin as a K-uplet $(x_1, \ldots, x_K)$, where x_i represents the number of element of size s_i in the bin.
As, $\forall i, x_i \leq n$, there are $O(n^K)$ possible bins.
A feasible solution requires at most n bins.
Thus, there are $O(n^{K+1})$ feasible packings.

Algorithm

Brute force all the feasible packings !

PTAS for BPP - All sizes $\geq \varepsilon$

Theorem

For all instances of the BPP where $\forall i, s_i \geq \varepsilon$, there is a $(1+\varepsilon)$-approximation algorithm

PTAS for BPP - All sizes $\geq \varepsilon$

Algorithm

1. Sort the sizes s_i such that $s_1 \leq s_2 \leq \cdots \leq s_n$ ($O(n \log n)$)
2. Partition the items into K groups, each with the same number of elements inside ($Q = \frac{n}{K}$) (except the last one)
3. Round every size in each group to the maximum size in the group ($O(n)$)
4. Brute force the solution for this new instance J (as seen before) ($O(n^{K+1})$)
5. Return the packing of J as a packing of I

PTAS for BPP - Finding K

Instances Memo

- I : Input instance
- J : I rounded up
- J': I rounded down
- J_Q: J without the last group
- J'_Q: J' without the first group

Computation of K

$$
\begin{aligned}
OPT(J) &\leq OPT(J_Q) + Q \leq OPT(J'_Q) + Q \\
&\leq OPT(J') + Q \leq OPT(I) + Q
\end{aligned}
$$

PTAS for BPP - Finding K

$$
\begin{aligned}
OPT(J) &\leq OPT(I) + Q \\
&\leq OPT(I) + \frac{n}{K} \\
&\leq OPT(I) + \frac{1}{K} \sum_{i=1}^{n} \frac{s_i}{\varepsilon} \\
&\leq OPT(I) + \frac{1}{K\varepsilon} \sum_{i=1}^{n} s_i \\
&\leq OPT(I) + \frac{1}{K\varepsilon} OPT(I) \\
&\leq \left(1 + \frac{1}{K\varepsilon}\right) OPT(I)
\end{aligned}
$$

PTAS for BPP - Finding K

Value of K

$$\frac{1}{\varepsilon K} = \varepsilon \implies K = \frac{1}{\varepsilon^2}$$

General Case

PTAS for BPP - General Case

Algorithm

1. Consider I' the instance obtained by keeping only from I (input instance) all the items with a size $\geq \varepsilon$
2. Solve I' with the $(1+\varepsilon)$-approximation algo $(O(n^{K+1}))$
3. Apply First Fit on the resulting packing using items with a size $< \varepsilon$ $(O(n))$
4. Return the packing

General case algorithm: $O(n^{K+1}) = O(n^{\frac{1}{\varepsilon^2}+1})$
If $\varepsilon = 0.01$, algorithm in $O(n^{10001})$

PTAS for BPP - General Case

Theorem

The previous algorithm finds a packing with at most $(1+2\varepsilon) \times OPT(I) + 1$ bins (for $\varepsilon \in \left]0, \frac{1}{2}\right]$)

Proof

Let B be the number of bins returned by the algorithm.

- If no extra bin is needed:
  $B \leq (1+2\varepsilon) \times OPT(I') \leq (1+2\varepsilon) \times OPT(I)$
- Else: The available space in each of the first $B-1$ bins is less than ε
  $$OPT(I) \geq \sum_i s_i > (B-1) \times (1-\varepsilon) \implies B < \frac{OPT(I)}{1-\varepsilon} + 1$$
  $$\frac{OPT(I)}{1-\varepsilon} + 1 \leq (1+2\varepsilon)OPT(I) + 1$$

Comparison

Comparison

$n = 20$

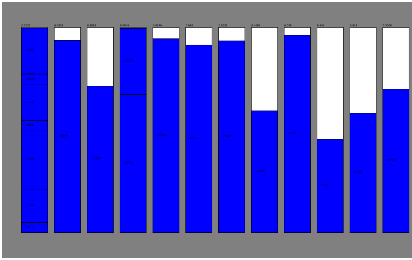

Figure: First Fit (12 bins)

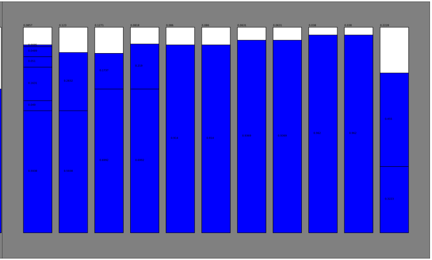

Figure: PTAS with $\varepsilon = 0.5$ (11 bins)