

# Approximationsalgorithmen

## Vorlesung 9: Approximationsschemata und das KNAPSACK-Problem

Folien von Joachim Spoerhase und Steven Chaplick

[Vazirani: §8, Williamson & Shmoys: §3.1]

# Approximation Schemes

Let  $\Pi$  be an optimization problem. An algorithm  $\mathcal{A}$  is called **polynomial-time approximation scheme (PTAS)**

if it outputs for every input  $(I, \epsilon)$  with  $I \in D_\Pi$  and  $\epsilon > 0$  a solution  $s \in S_\Pi(I)$  such that

- $\text{obj}_\Pi(I, s) \geq (1 - \epsilon) \cdot \text{OPT}$  if  $\Pi$  max problem.
- $\text{obj}_\Pi(I, s) \leq (1 + \epsilon) \cdot \text{OPT}$  if  $\Pi$  min problem,

The runtime of  $\mathcal{A}$  is polynomial in  $|I|$  for every fixed  $\epsilon > 0$ .

A PTAS is called **fully polynomial-time approximation scheme (FPTAS)** if its runtime is polynomial in  $|I|$  and  $1/\epsilon$ .

Example running times:

- $O(n^{1/\epsilon^2}) \rightsquigarrow$  PTAS
- $O(2^{1/\epsilon} n^4) \rightsquigarrow$  PTAS
- $O((\frac{1}{\epsilon})^2 n^3) \rightsquigarrow$  FPTAS (and PTAS ;-)

# KNAPSACK Problem

We are given a set  $S = \{a_1, \dots, a_n\}$  of **objects**.

For  $i = 1, \dots, n$ , object  $a_i$  has **size**  $\text{size}(a_i) \in \mathbb{N}^+$  and **profit**  $\text{profit}(a_i) \in \mathbb{N}^+$ . We are also given a (knapsack) **capacity**  $B \in \mathbb{N}^+$ . We are looking for a subset of the objects whose **total size** is at most  $B$  and whose **total profit** is maximum.

NP-hard

# Pseudopolynomial-Time Algorithm

Let  $\Pi$  be an optimization problem whose instances can be represented by **objects** (such as sets, elements, edges, nodes) and **numbers** (such as costs, weights, profits). As usual,  $|I|$  denotes the size of the instance  $I \in D_{\Pi}$ , where all numbers in  $I$  are encoded in **binary**. We use  $|I|_u$  to denote the size of  $I$  when all numbers in  $I$  are encoded in **unary**.

- The running time of a polynomial-time algorithm for  $\Pi$  is polynomial in  $|I|$ .
- The running time of a **pseudo-polynomial-time algorithm** is polynomial in  $|I|_u$ .
- The running time of a pseudo-polynomial algorithm may not be polynomial in  $|I|$ .

# Pseudo-Polynomial Alg. for KNAPSACK

- $P := \max_i \text{profit}(a_i) \Rightarrow P \leq \text{OPT} \leq nP$
- For every  $i = 1, \dots, n$  and every  $p \in \{1, \dots, nP\}$ , let  $S_{i,p}$  be a subset of  $\{a_1, \dots, a_i\}$  whose total profit is precisely  $p$  and whose total size is minimum among all subsets with these properties. Such a set may not exist.
- Let  $A[i, p]$  denote the total size of  $S_{i,p}$  (let  $A[i, p] = \infty$  if no such set exists).
- If all entries of the table  $A[\cdot, \cdot]$  are known, we can compute  $\text{OPT} = \max\{p \mid A(n, p) \leq B\}$ .

# Pseudo-Polynomial Alg. for KNAPSACK

- $A[1, p]$  can easily be determined for  $p \in \{0, \dots, nP\}$ .
- For  $i = 1, \dots, n$ , set  $A[i, p] := \infty$  for  $p < 0$  and  $A[i, 0] = 0$ .
- $A[i + 1, p] = \min\{ A[i, p], \text{size}(a_{i+1}) + A[i, p - \text{profit}(a_{i+1})] \}$
- Running time and space consumption:  
 $A[\cdot, \cdot]$  and OPT can be computed in  $O(n^2P)$  time & space.

KNAPSACK can be solved to optimality in  $O(n^2P)$  time.

# FPTAS for KNAPSACK by Scaling

Note:  $O(n^2P)$  is polynomial in  $n$  – if  $P$  is polynomial in  $n$  :-)

- FPTAS-Idea: **Scale** profits to polynomial size (as required by the error parameter  $\epsilon$ ).

# FPTAS for KNAPSACK by Scaling

KnapsackScaling( $I, \epsilon$ )

$$K \leftarrow \epsilon P / n$$

$$\text{profit}'(a_i) := \lfloor \text{profit}(a_i) / K \rfloor$$

compute optimum solution  $S'$  for  $I$  w.r.t.  $\text{profit}'(\cdot)$

**return**  $S'$

**Lemma.**  $\text{profit}(S') \geq (1 - \epsilon) \cdot \text{OPT}$ .

*Proof.* Let  $o_1, \dots, o_k$  be an optimal solution.

**Obs. 1.** For  $i = 1, \dots, k$ ,  $\text{profit}(o_i) - K \leq K \cdot \text{profit}'(o_i) \leq \text{profit}(o_i)$

$$\Rightarrow \text{OPT} - \epsilon P = \text{OPT} - nK \leq \text{OPT} - kK \leq K \sum_i \text{profit}'(o_i)$$

**Obs. 2.**  $\text{profit}(S') \geq K \cdot \text{profit}'(S') \geq K \cdot \sum_i \text{profit}'(o_i)$

$$\Rightarrow \text{profit}(S') \geq \text{OPT} - \epsilon P \geq \text{OPT} - \epsilon \text{OPT} = (1 - \epsilon) \cdot \text{OPT} \quad \square$$

**Thm. 1.** KnapsackScaling is an FPTAS for KNAPSACK.  
The algorithm runs in  $O(n^3/\epsilon)$  time. [Ibarra & Kim, '75]



# Strong NP-Hardness

An optimization problem is called **strongly NP-hard** if it remains NP-hard when numbers are encoded in unary.

Can the `KNAPSACK` problem be strongly NP-hard?

**Thm. 2.** A strongly NP-hard problem has no pseudo-polynomial algorithm unless  $P = NP$ .

# FPTAS and Pseudo-Polynomial Algorithms

**Thm. 3.** Let  $p$  be a polynomial and let  $\Pi$  be an NP-hard minimization problem with integral objective function and with  $\text{OPT}(I) < p(|I|_u)$  for all instances  $I$  of  $\Pi$ . If  $\Pi$  has an FPTAS then there is a pseudo-polynomial algorithm for  $\Pi$ .

*Proof.*

Suppose there is an FPTAS for  $\Pi$  (running in  $q(|I|, 1/\epsilon)$  time).

Set  $\epsilon = 1/p(|I|_u)$ .

$\Rightarrow \text{ALG} \leq (1 + \epsilon)\text{OPT} < \text{OPT} + \epsilon p(|I|_u) = \text{OPT} + 1.$

$\Rightarrow \text{ALG} = \text{OPT}.$

Running time:  $q(|I|, p(|I|_u))$ , that is, polynomial in  $|I|_u$ .

# FPTAS and Strong NP-Hardness

**Corollary.** Let  $\Pi$  be an NP-hard optimization problem that fulfils the requirements of Theorem 3.  
If  $\Pi$  is strongly NP-hard then there is no FPTAS for  $\Pi$  unless  $P = NP$ .