Approximation Algorithms

Lecture 8: Approximation Schemes and the Knapsack Problem

Part I: Knapsack

Joachim Spoerhase

Winter 2021/22

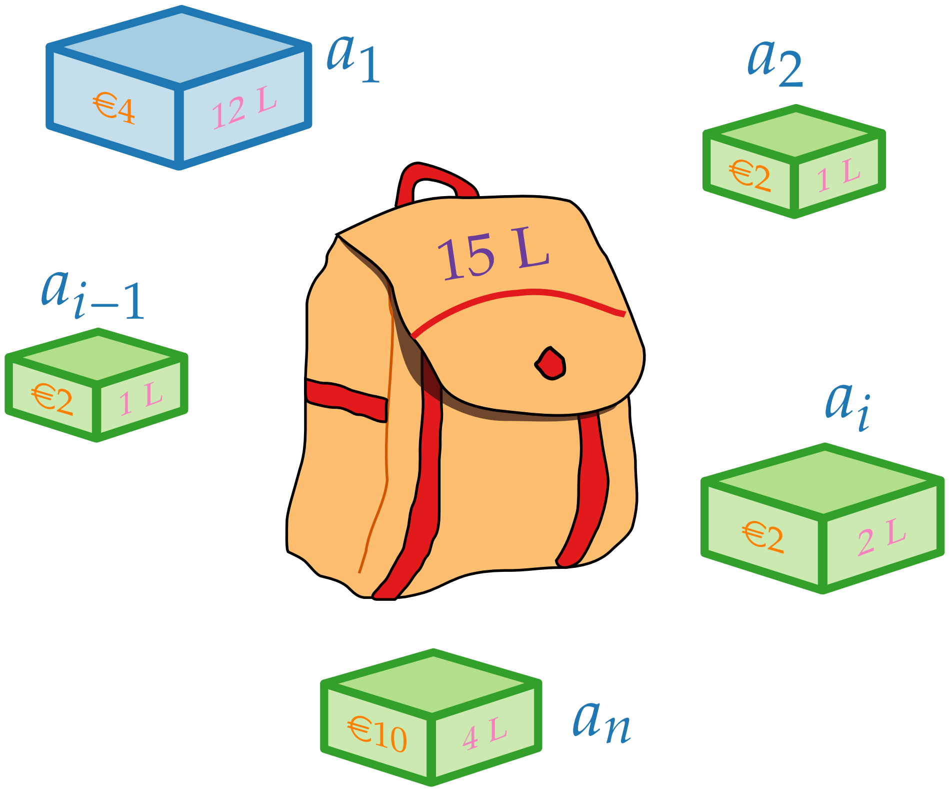
Knapsack

Given:

- A set $S = \{a_1, \ldots, a_n\}$ of **objects**.
- For every object a_i a **size** $\text{size}(a_i) \in \mathbb{N}^+$
- For every object a_i a **Profit** $\text{profit}(a_i) \in \mathbb{N}^+$
- A knapsack **capacity** $B \in \mathbb{N}^+$

Task: Find a subset of objects whose **total size** is at most B and whose **total profit** is maximum.

NP-hard



Approximation Algorithms

Lecture 8:
Approximation Schemes and the Knapsack Problem

Part II:
Pseudo-Polynomial Algorithms and Strong NP-Hardness

Joachim Spoerhase

Winter 2021/22

Pseudo-Polynomial Algorithms

Let Π be an optimization problem whose instances can be represented by **objects** (such as sets, elements, edges, nodes) and **numbers** (such as costs, weights, profits).

$|I|$: The size of an instance $I \in D_\Pi$, where all numbers in I are encoded in **binary**. $(5 \mathrel{\hat{=}} 101 \Rightarrow |I| = 3)$

$|I|_\mathrm{u}$: The size of an instance $I \in D_\Pi$, where all numbers in I are encoded in **unary**. $(5 \mathrel{\hat{=}} 11111 \Rightarrow |I|_\mathrm{u} = 5)$

The running time of a polynomial algorithm for Π is polynomial in $|I|$.

The running time of a **pseudo-polynomial algorithm** is polynomial in $|I|_\mathrm{u}$.

The running time of a pseudo-polynomial algorithm may not be polynomial in $|I|$.

Strong NP-Hardness

An optimization problem is called **strongly NP-hard** if it remains NP-hard under unary encoding.

An optimization problem is called **weakly NP-hard** if it is NP-hard under binary encoding but has a pseudo-polynomial algorithm.

Theorem. A strongly NP-hard problem has no pseudo-polynomial algorithm unless P = NP.

Approximation Algorithms

Lecture 8: Approximation Schemes and the Knapsack Problem

Part III: Pseudo-Polynomial Algorithm for Knapsack

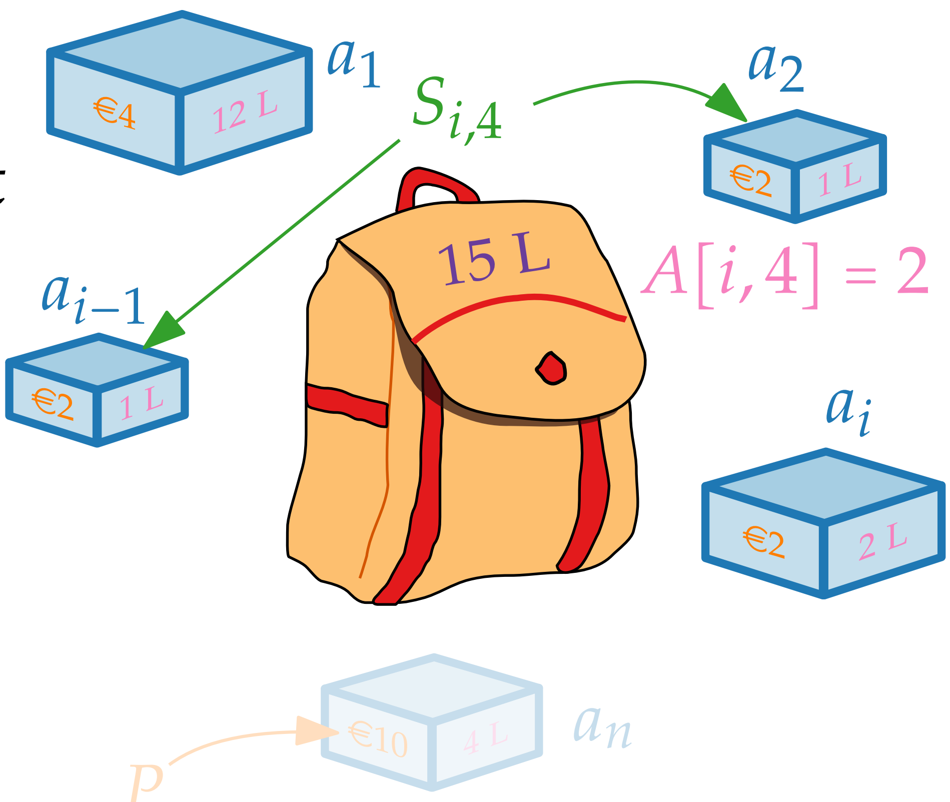
Joachim Spoerhase

Winter 2021/22

Pseudo-Polynomial Alg. for KNAPSACK

Let $P := \max_i \text{profit}(a_i) \quad \Rightarrow \quad P \leq \text{OPT} \leq nP$

For every $i = 1, \ldots, n$ and every $p \in \{1, \ldots, nP\}$, let $S_{i,p}$ be a subset of $\{a_1, \ldots, a_i\}$ 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 $A[i, p]$ be the total size of set $S_{i,p}$ (set $A[i, p] = \infty$ if no such set exists).

If all $A[i, p]$ are known, then we can compute $\text{OPT} = \max\{\, p \mid A[n, p] \leq B \,\}$.

Pseudo-Polynomial Alg. for KNAPSACK

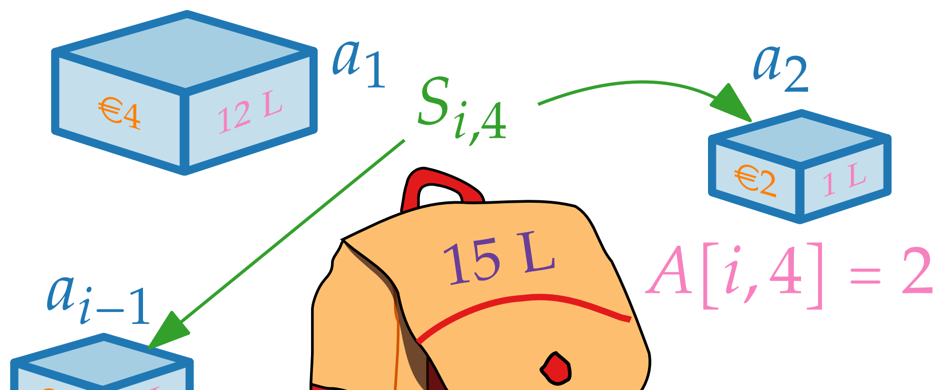
$A[1,p]$ can be computed for all $p \in \{0,\dots,nP\}$.

Set $A[i,p] := \infty$ for $p < 0$

$A[i+1,p] = \min\{A[i,p], \mathrm{size}(a_{i+1}) + A[i, p - \mathrm{profit}(a_{i+1})]\}$

$\Rightarrow$ All values $A[i,p]$ can be computed in total time $O(n^2 P)$.

$\Rightarrow$ OPT can be computed in $O(n^2 P)$ time.



Theorem. KNAPSACK can be solved optimally in pseudo-polynomial time $O(n^2 P)$.

Corollary. KNAPSACK is weakly NP-hard.

Pseudo-Polynomial Alg. for KNAPSACK

Theorem. KNAPSACK can be solved optimally in pseudo-polynomial time $O(n^2 P)$.

Examples. $P = n^5$ $\Rightarrow$ running time $O(n^7)$

(Bin.) instance size $|I| \geq n \log P = \Omega(n \log n)$

$\Rightarrow n \in O(|I| / \log |I|)$

$\Rightarrow$ running time $O(|I|^7 / \log^7 |I|) = \text{poly}(|I|)$

$P = 2^n$ $\Rightarrow$ running time $O(n^2 2^n)$

(Bin.) instance size $|I| \leq n \log P = O(n^2)$

$\Rightarrow$ running time $O(|I| 2^{\sqrt{|I|}}) \neq \text{poly}(|I|)$

(Un.) instance size $|I|_u \leq nP = O(n 2^n)$

$\Rightarrow n \in O(\log |I|_u - \log \log |I|_u)$

$\Rightarrow$ running time $O(|I|_u \log |I|_u) = \text{poly}(|I|_u)$

Observe. Running time $O(n^2 P)$ poly in n if P poly in n.

Approximation Algorithms

Lecture 8: Approximation Schemes and the Knapsack Problem

Part IV: Approximation Schemes

Joachim Spoerhase

Winter 2021/22

Approximation Schemes

Let Π be an optimization problem. An algorithm $\mathcal{A}$ is called a **polynomial-time approximation scheme** (PTAS), if it outputs for every input (I, ε) with $I \in D_\Pi$ and $\varepsilon > 0$ a solution $s \in S_\Pi(I)$ such that the following holds:

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

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

$\mathcal{A}$ is called **fully polynomial-time approximation scheme** (FPTAS) if its running time is polynomial in $|I|$ and $1/\varepsilon$.

Example running times

- $O(n^{1/\varepsilon}) \rightsquigarrow$ PTAS
- $O(2^{1/\varepsilon} n^4) \rightsquigarrow$ PTAS
- $O(n^3/\varepsilon^2) \rightsquigarrow$ FPTAS

Approximation Algorithms

Lecture 8:
Approximation Schemes and the Knapsack Problem

Part V:
FPTAS for Knapsack

Joachim Spoerhase

Winter 2021/22

FPTAS for KNAPSACK via Scaling

KnapsackScaling (I, ε)

$K \leftarrow \varepsilon P/n$ // scaling factor
$\text{profit}'(a_i) := \lfloor \text{profit}(a_i)/K \rfloor$
Compute optimal solution S' for I w.r.t. $\text{profit}'(\cdot)$
return S'

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

Proof. Let $\text{OPT} = \{o_1, \dots, o_k\}$.

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

$\Rightarrow K \cdot \sum_i \text{profit}'(o_i) \geq \text{OPT} - kK \geq \text{OPT} - nK = \text{OPT} - \varepsilon P$

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

$\Rightarrow \text{profit}(S') \geq \text{OPT} - \varepsilon P \geq \text{OPT} - \varepsilon\, \text{OPT} = (1-\varepsilon) \cdot \text{OPT}$ □

Theorem. KnapsackScaling is an FPTAS for KNAPSACK with running time $O(n^3/\varepsilon) = O\left(n^2 \cdot \frac{P}{\varepsilon P/n}\right)$.

Approximation Algorithms

Lecture 8: Approximation Schemes and the Knapsack Problem

Part VI: Connections

Joachim Spoerhase

Winter 2021/22

FPTAS and Pseudo-Poly. Algorithms

Theorem. Let p be a polynomial and let Π be an NP-hard minimization problem with integral objective function and $\mathrm{OPT}(I) < p(|I|_{\mathrm{u}})$ for all instances I of Π. If Π has an FPTAS, then there is a pseudo-polynomial algorithm for Π.

Proof.

Assuming there is an FPTAS for Π (in $q(|I|, 1/\varepsilon)$ time).

Set $\varepsilon = 1/p(|I|_{\mathrm{u}})$.

$\Rightarrow \mathrm{ALG} \leq (1+\varepsilon)\mathrm{OPT} < \mathrm{OPT} + \varepsilon p(|I|_{\mathrm{u}}) = \mathrm{OPT} + 1.$

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

Running time: $q(|I|, p(|I|_{\mathrm{u}}))$, so $\mathrm{poly}(|I|_{\mathrm{u}})$.

FPTAS and Strong NP-Hardness

Theorem. A strongly NP-hard problem has no pseudo-polynomial algorithm unless P = NP.

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

Corollary. Let Π be an NP-hard optimization problem that fulfils the restrictions above. If Π is strongly NP-hard, then there is no FPTAS for Π (unless P = NP).