Approximation Algorithms

Lecture 8: Approximation Schemes and the KNAPSACK Problem

Part I: KNAPSACK

Alexander Wolff

Winter 2023/24

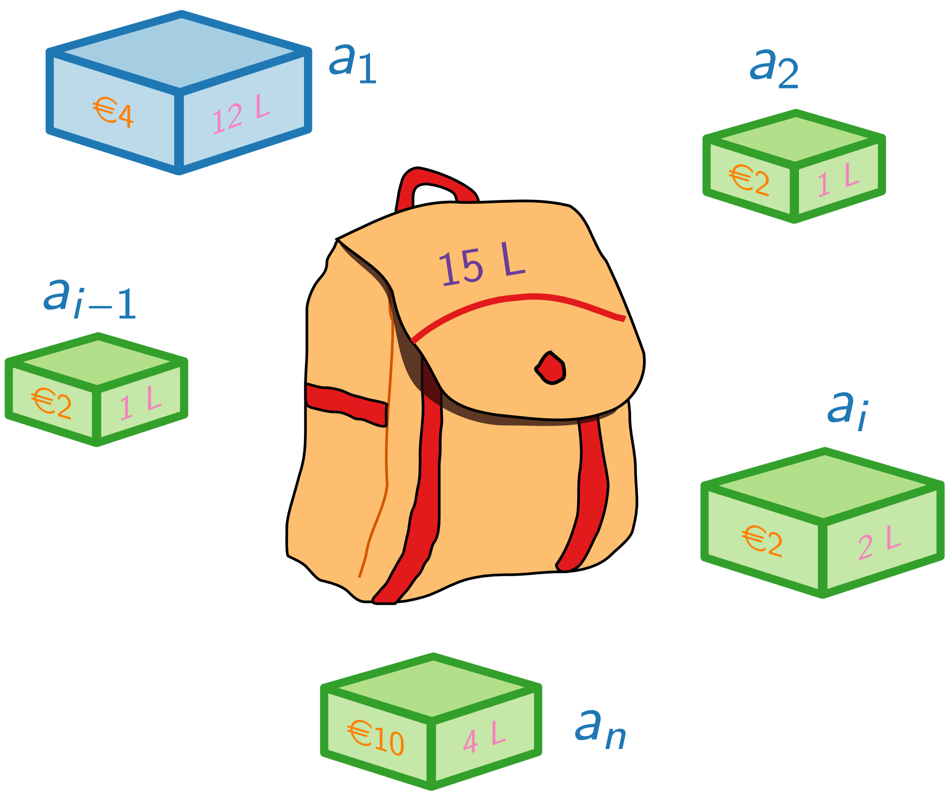
Knapsack

Given:

- A set $S = \{a_1, \dots, a_n\}$ 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

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**. $(5 \triangleq 101_\mathrm{b} \Rightarrow |I| = 3)$

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

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

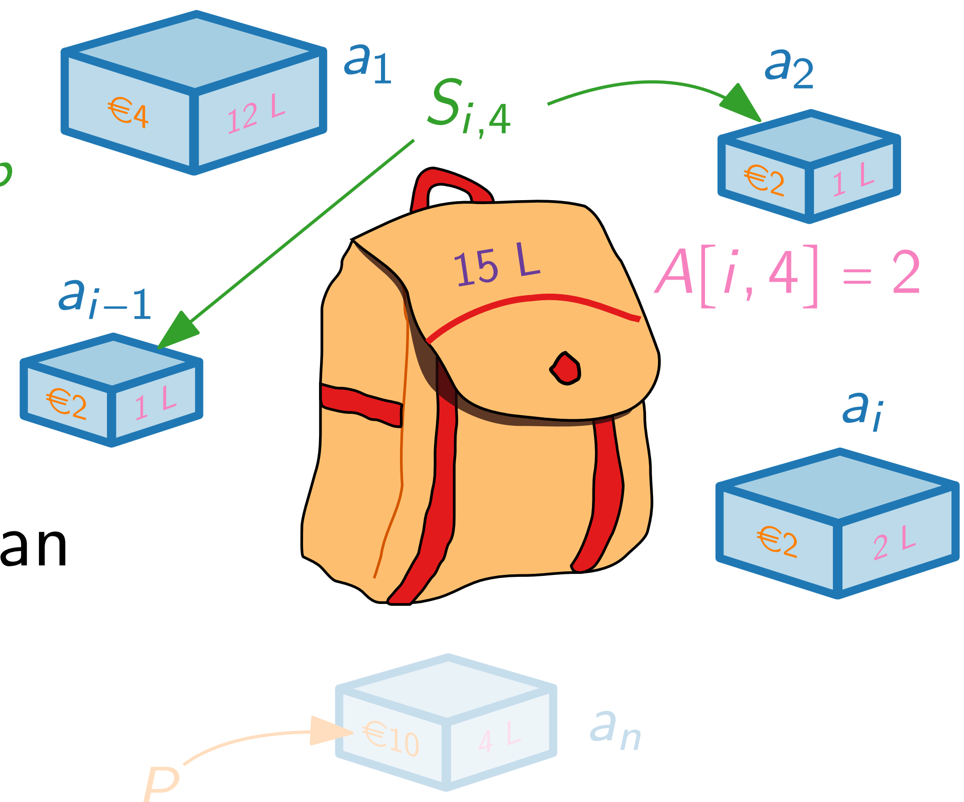
Part III: Pseudo-Polynomial Algorithm for KNAPSACK

Pseudo-Polynomial Alg. for KNAPSACK

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

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 $S_{i,p}$ (set $A[i,p] = \infty$ if no such set exists).



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

Pseudo-Polynomial Alg. for Knapsack

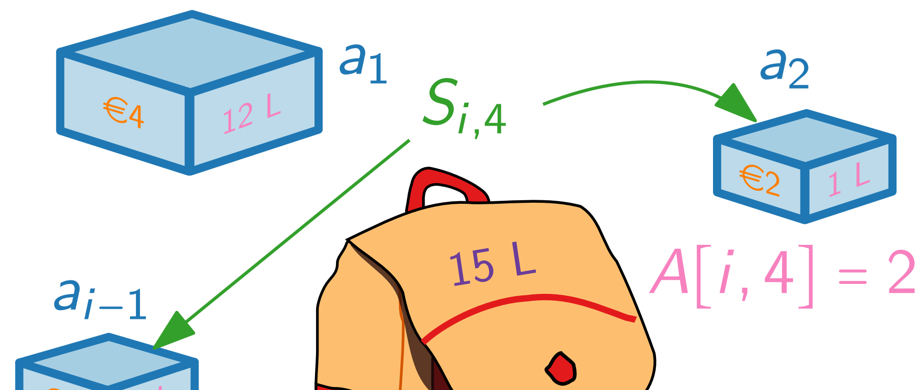
$A[1, p]$ can be computed for every $p \in \{0, \dots, nP\}$.

Set $A[i, p] := \infty$ for $p < 0$ (for convenience).

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

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

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



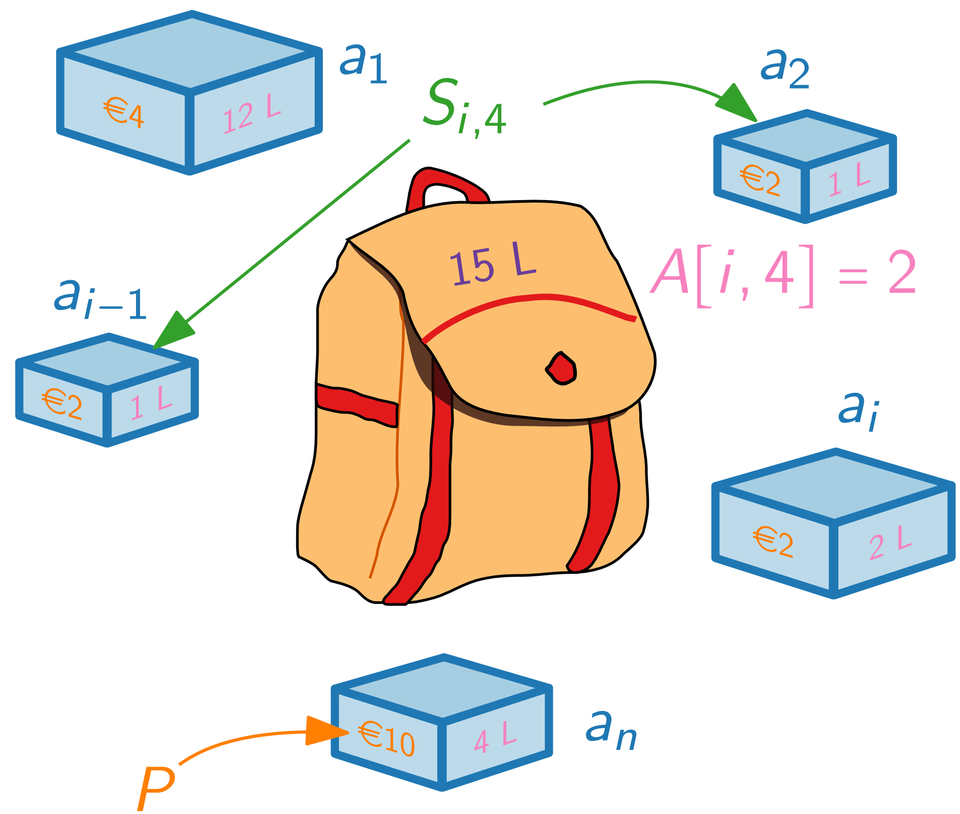
Theorem. Knapsack can be solved optimally in pseudo-polynomial time $O(n^2 P)$.

Corollary. Knapsack is weakly NP-hard.

Pseudo-Polynomial Alg. for Knapsack

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

Observe. The running time $O(n^2 P)$ is polynomial in n if P is polynomial in n.



Approximation Algorithms

Lecture 8:
Approximation Schemes and the KNAPSACK Problem

Part IV:
Approximation Schemes

Approximation Schemes

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

- $\mathrm{obj}_\Pi(I, s) \le (1 + \varepsilon) \cdot \mathrm{OPT}$ if Π is a minimization problem,
- $\mathrm{obj}_\Pi(I, s) \ge (1 - \varepsilon) \cdot \mathrm{OPT}$ if Π is a maximization problem,

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

Approximation Algorithms

Lecture 8: Approximation Schemes and the Knapsack Problem

Part V: FPTAS for Knapsack

An FPTAS for KNAPSACK via Scaling

KnapsackScaling (I, ε)

$K = \varepsilon P/n$ // 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 $\text{OPT} = \{o_1, \dots, o_\ell\}$.

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

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

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

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

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 Between the Concepts

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.

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

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

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

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

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 fulfills the restrictions above.
If Π is strongly NP-hard, then there is no FPTAS for Π (unless P = NP).