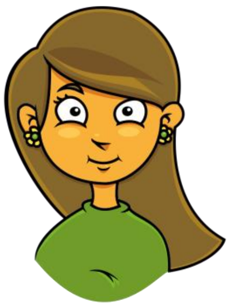
The Subset Sum Game Revisited

Astrid Pieterse[1] and Gerhard J. Woeginger[2]

[1] Eindhoven University of Technology
[2] RWTH Aachen

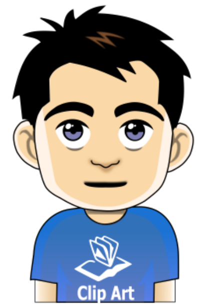
The Subset Sum Game

Alice



Items $a_1, \ldots, a_n$

Bob



Items $b_1, \ldots, b_m$

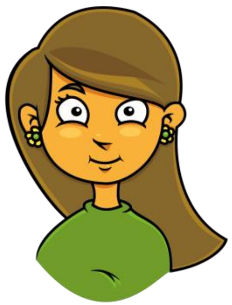
Alice: 6, 6, 11

Bob: 4, 4, 7, 4, 7

Knapsack: 24

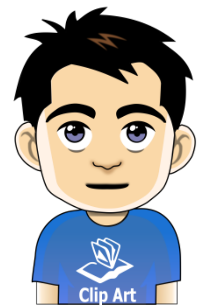
The Subset Sum Game

Alice

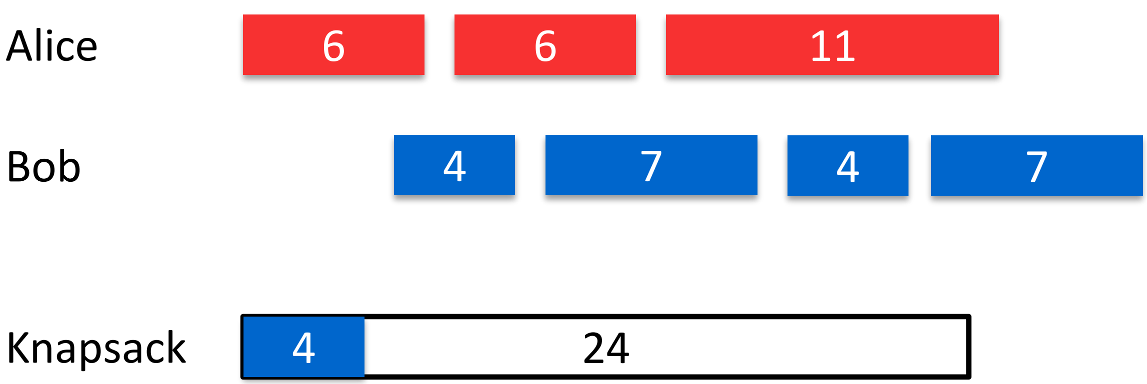


Items $a_1, \ldots, a_n$

Bob

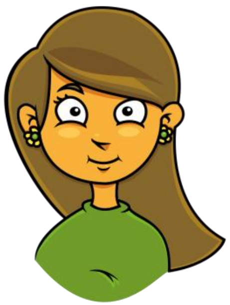


Items $b_1, \ldots, b_m$



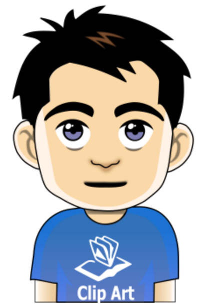
The Subset Sum Game

Alice

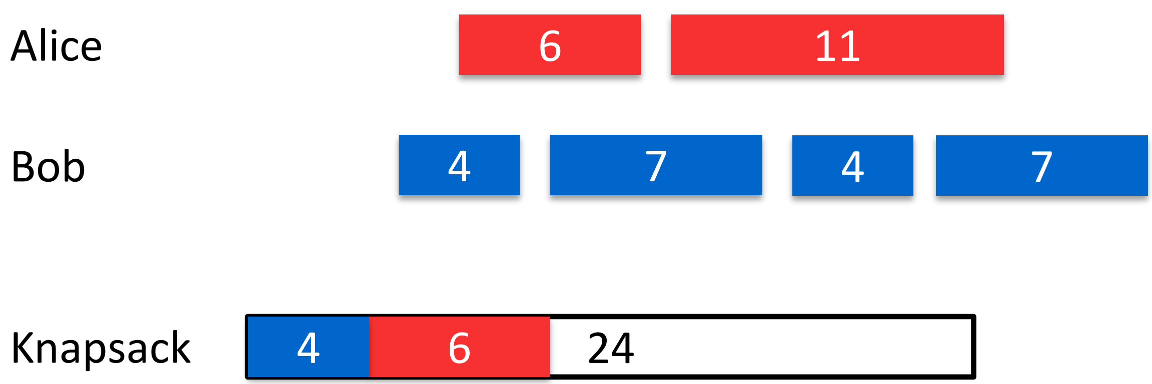


Items $a_1, \dots, a_n$

Bob

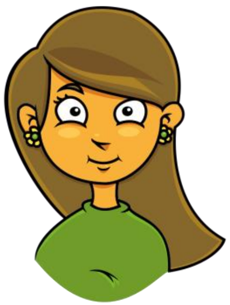


Items $b_1, \dots, b_m$



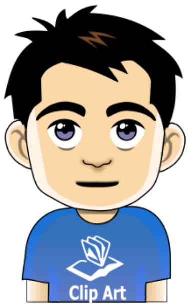
The Subset Sum Game

Alice

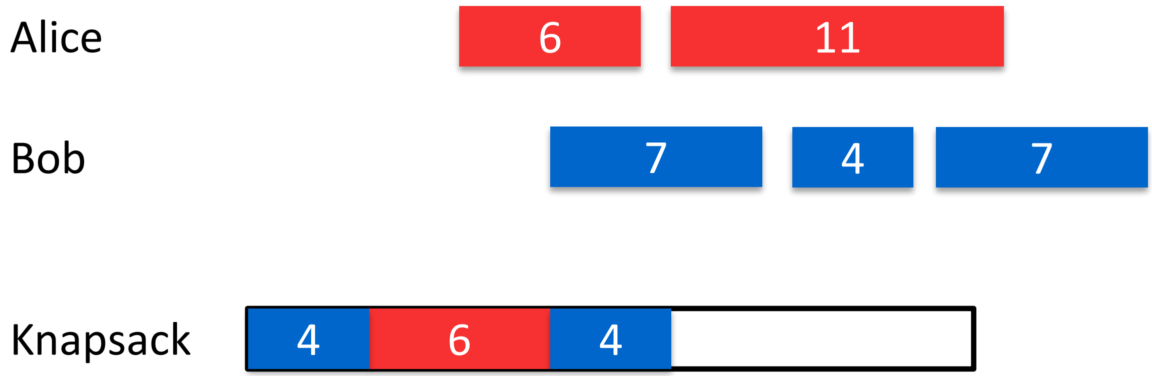


Items $a_1, \ldots, a_n$

Bob

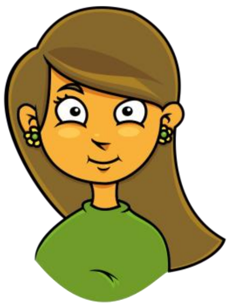


Items $b_1, \ldots, b_m$



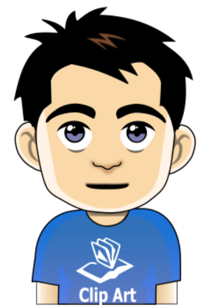
The Subset Sum Game

Alice

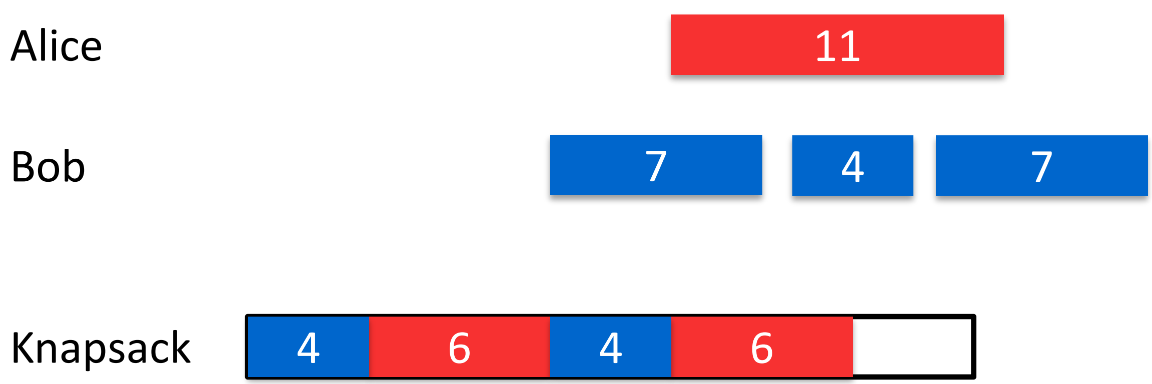


Items $a_1, \dots, a_n$

Bob

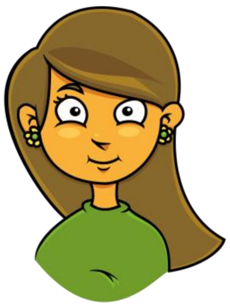


Items $b_1, \dots, b_m$



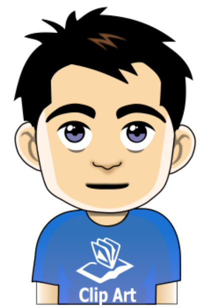
The Subset Sum Game

Alice

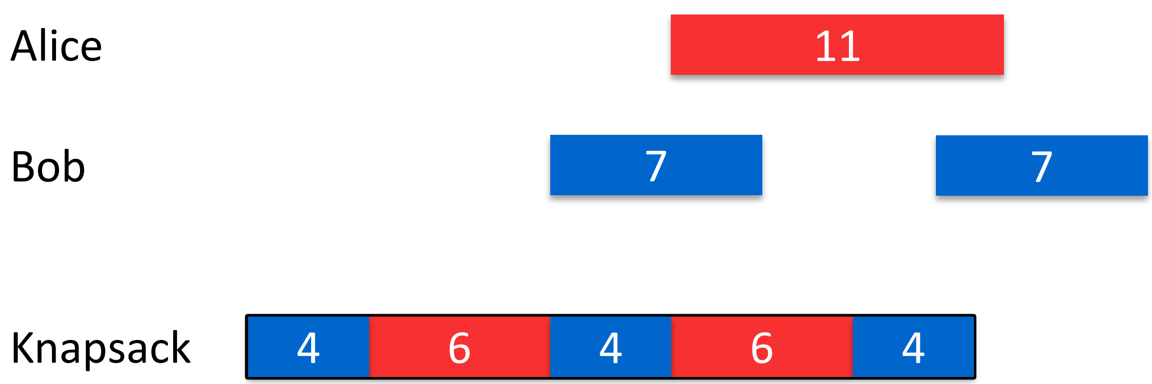


Items $a_1, \dots, a_n$

Bob

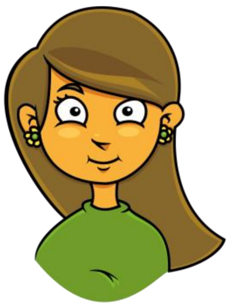


Items $b_1, \dots, b_m$



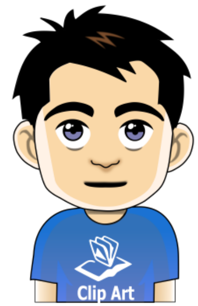
The Subset Sum Game

Alice

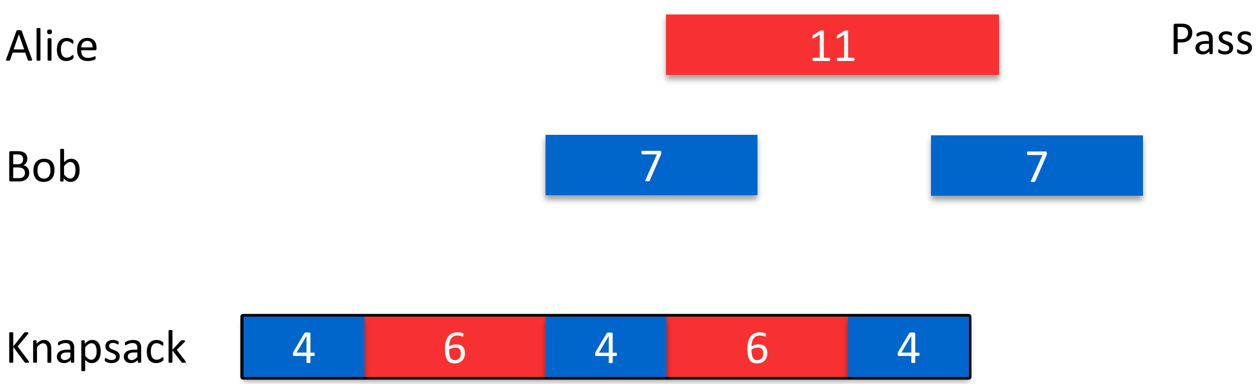


Items $a_1, \dots, a_n$

Bob

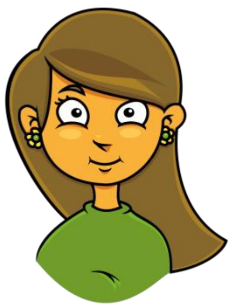


Items $b_1, \dots, b_m$



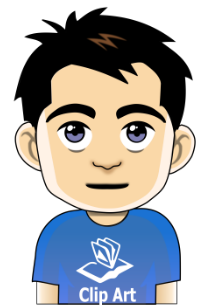
The Subset Sum Game

Alice

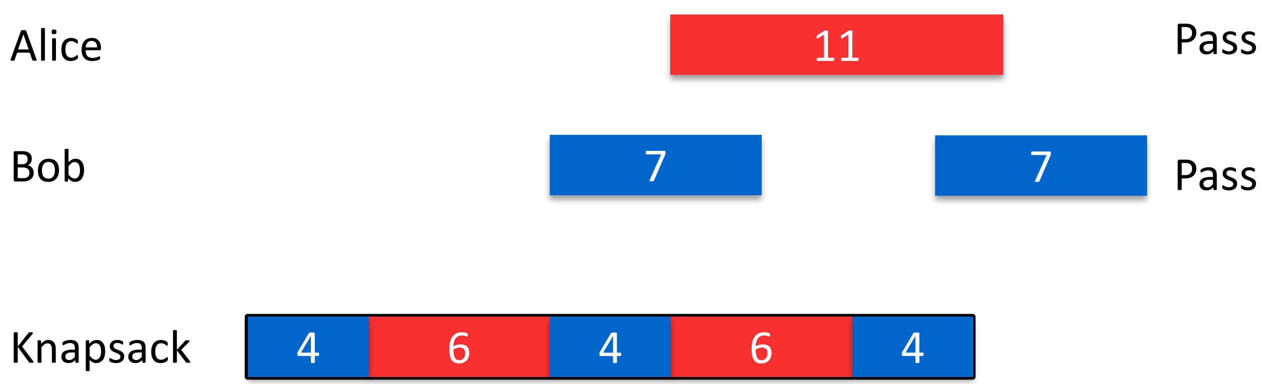


Items $a_1, \ldots, a_n$

Bob

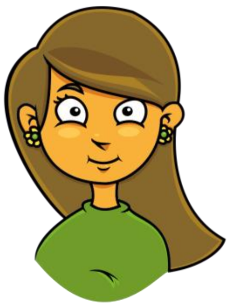


Items $b_1, \ldots, b_m$



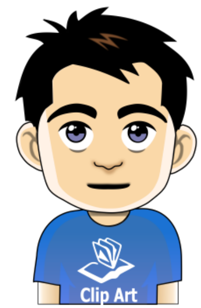
The Subset Sum Game

Alice



Items $a_1, \dots, a_n$

Bob



Items $b_1, \dots, b_m$

Alice: 12

Bob: 12

Alice 11 Pass

Bob 7 7 Pass

Knapsack 4 6 4 6 4

Motivation

Players compete over a resource

- Processing time
- Bandwidth
- …

Player strategies

Players know

- Other player's goal
- Items

We will be Alice

- Goal: Maximize our weight (Selfish)

Strategies for Bob

Strategies for Bob

- Selfish

Result A

Result B

- Hostile
- Greedy

Strategies for Bob

Strategies for Bob

- Selfish

	Result A
Alice	6
Bob	4

	Result B
Alice	7
Bob	6

- Hostile

- Greedy

Strategies for Bob

Strategies for Bob

- Selfish

	Result A	Result B
Alice	6	7
Bob	4	6

- Hostile
- Greedy

Strategies for Bob

Strategies for Bob

- Selfish

	Result A		Result B
Alice	6	Alice	7
Bob	4	Bob	6

- Hostile

Alice	6	Alice	7
Bob	4	Bob	6

- Greedy

Strategies for Bob

Strategies for Bob

- Selfish

	Result A		Result B
Alice	6	Alice	7
Bob	4	Bob	6

- Hostile

Alice	6	Alice	7
Bob	4	Bob	6

- Greedy

Strategies for Bob

Strategies for Bob

- Selfish

	Result A		Result B
Alice	6	Alice	7
Bob	4	Bob	6

- Hostile

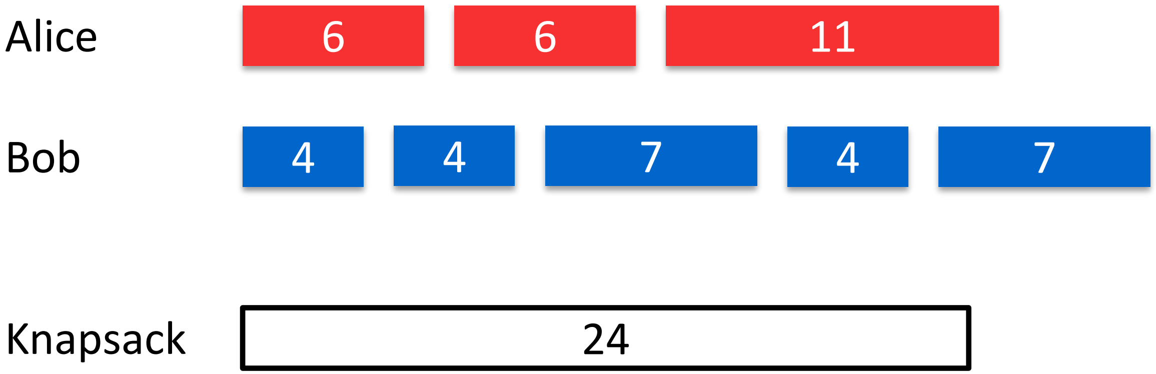
Alice	6	Alice	7
Bob	4	Bob	6

- Greedy
  - Play the largest item that fits

Hostile – Selfish

Alice plays selfish

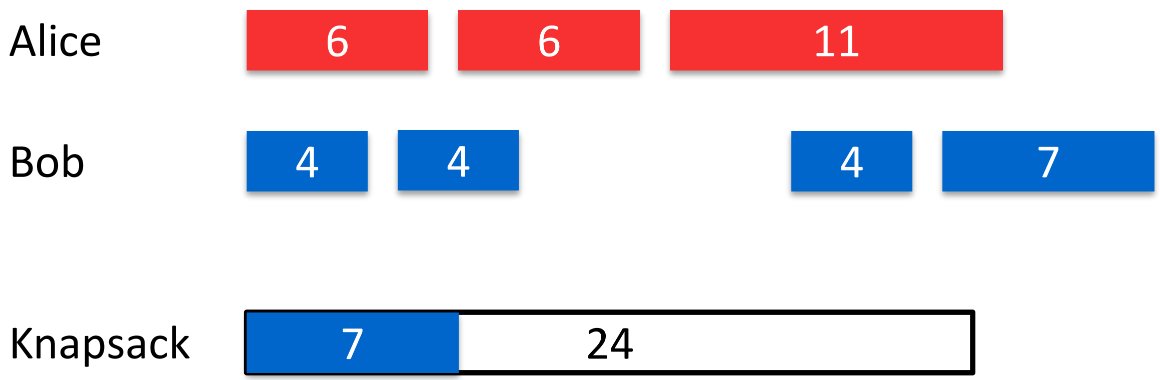
- Does it matter if Bob is selfish or hostile?



Hostile – Selfish

Alice plays selfish

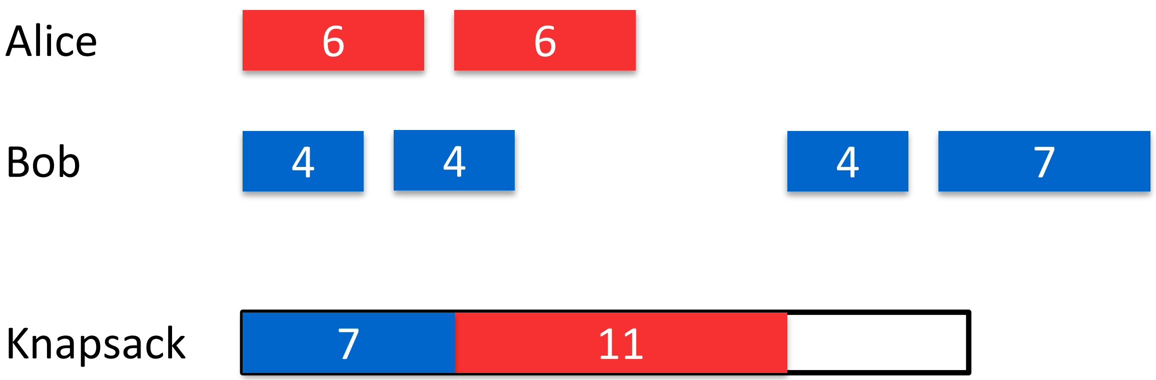
- Does it matter if Bob is selfish or hostile?



Hostile – Selfish

Alice plays selfish

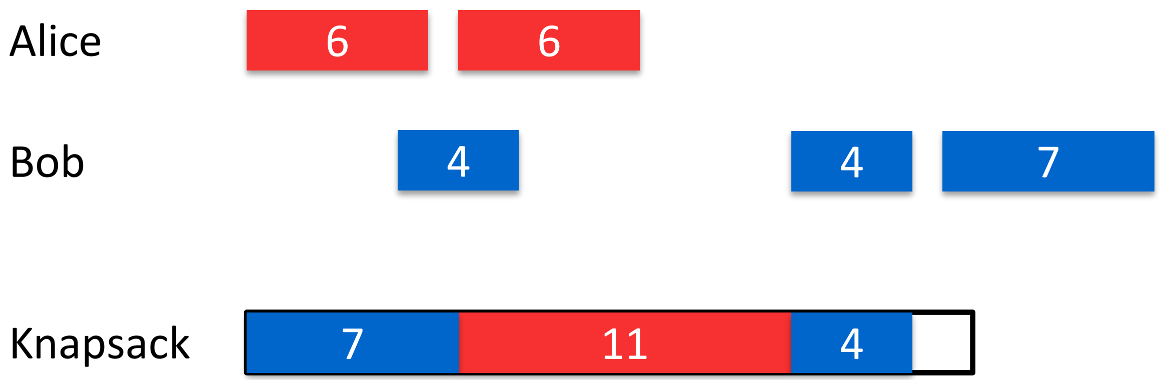
- Does it matter if Bob is selfish or hostile?



Hostile – Selfish

Alice plays selfish

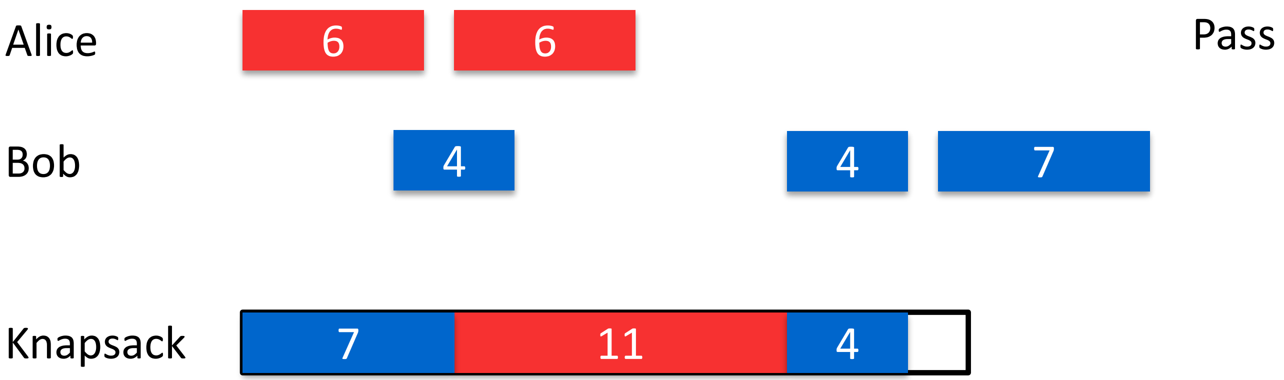
- Does it matter if Bob is selfish or hostile?



Hostile – Selfish

Alice plays selfish

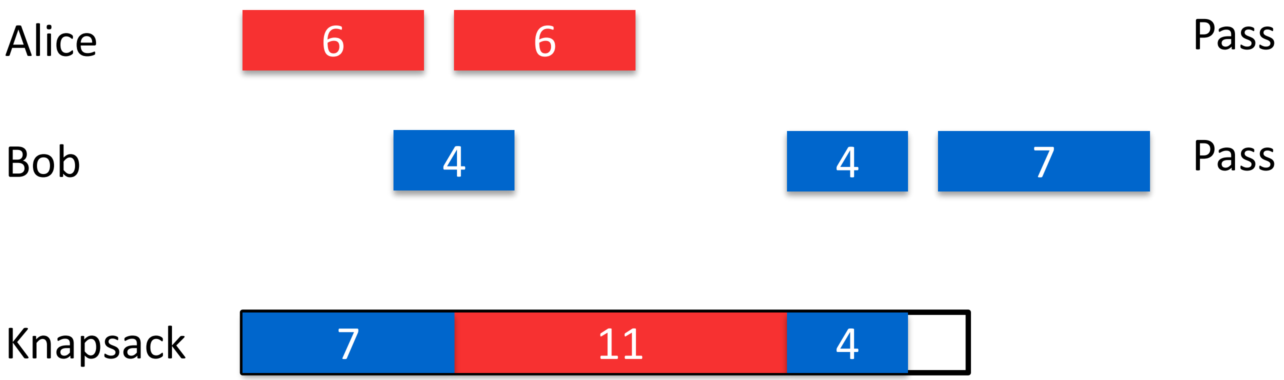
- Does it matter if Bob is selfish or hostile?



Hostile – Selfish

Alice plays selfish

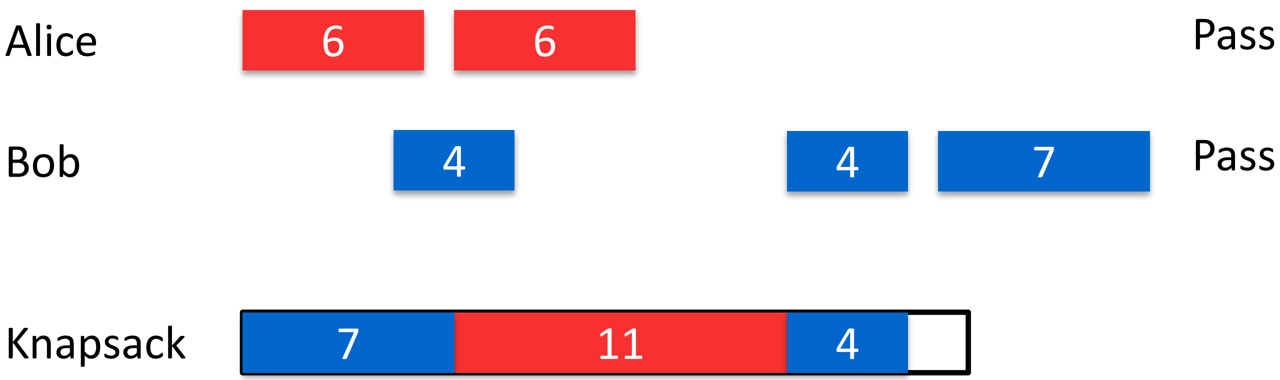
- Does it matter if Bob is selfish or hostile?



Hostile – Selfish

Alice plays selfish

- Does it matter if Bob is selfish or hostile?

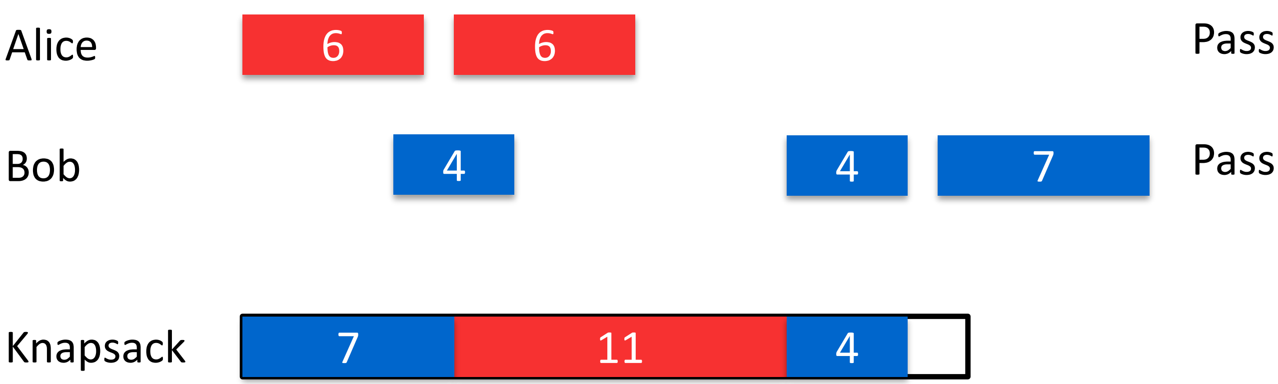


Result: Alice: 11, Bob: 11

Hostile – Selfish

Alice plays selfish

- Does it matter if Bob is selfish or hostile?



Result: Alice: 11, Bob: 11

Previously: Alice: **12**, Bob: **12**

Previous work

Introduced by

Darmann, Nicosia, Pferschy & Schauer

Greedy strategy [Darmann et al.]

- Gives at least 1/2 the maximum weight
- Greedy with lookahead

Optimal strategy against selfish Bob may pack smaller items before larger items [Darmann et al.]

- But not against greedy
- Packs items in non-increasing order

Our results

SSG against hostile/selfish

- $PSPACE$ complete
- No α-approximation unless $P = NP$

SSG against greedy

- Solvable in pseudo polynomial time
- Has a PTAS
- Has no FPTAS unless $P = NP$

PTAS against greedy player

Definition: PTAS

Polynomial time approximation scheme

- Trade-off: running time - approximation factor

Algorithm that for given $\varepsilon > 0$

- Finds strategy for Alice
  - with value at least $(1 - \varepsilon) * OPT$
- Runs in polynomial time in n for ε constant
  - $O(n^{1/\varepsilon})$ allowed

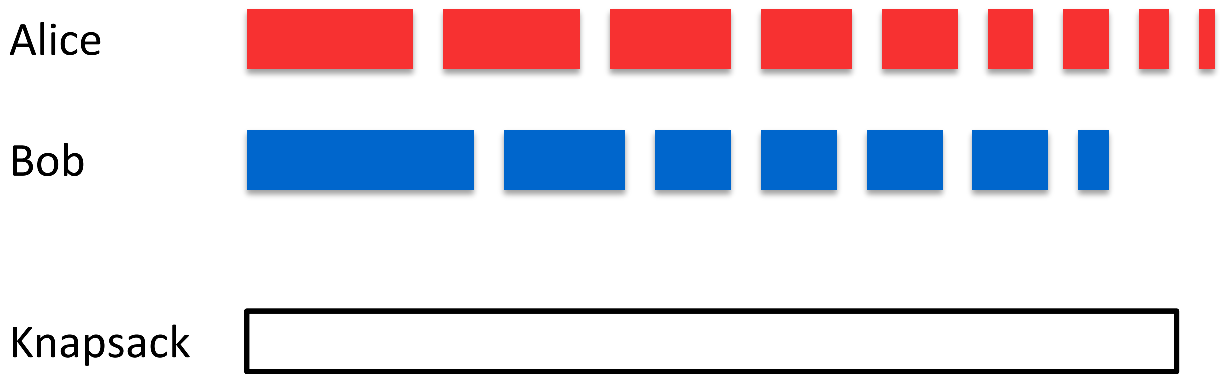
PTAS against greedy

Idea

- Pack in non-increasing order
- Try all possibilities to pack the first few items
  - Large items are important
- Pack smaller items greedily
- Choose the best strategy

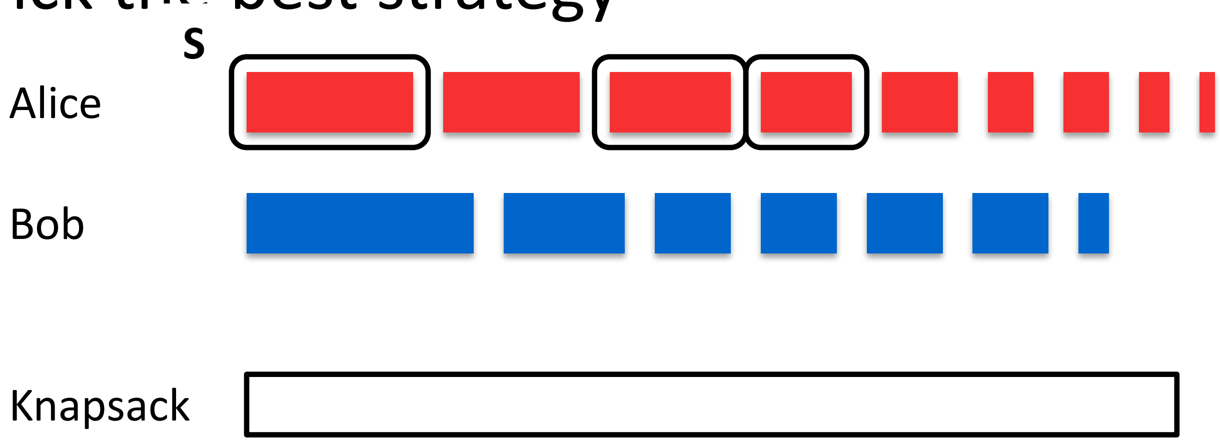
PTAS

- For each $S \subseteq$ Alice-items with $|S| \leq 1/\varepsilon$
  - Choose items of S large to small
  - Continue greedily
- If impossible to pack S
  - Discard
- Pick the best strategy



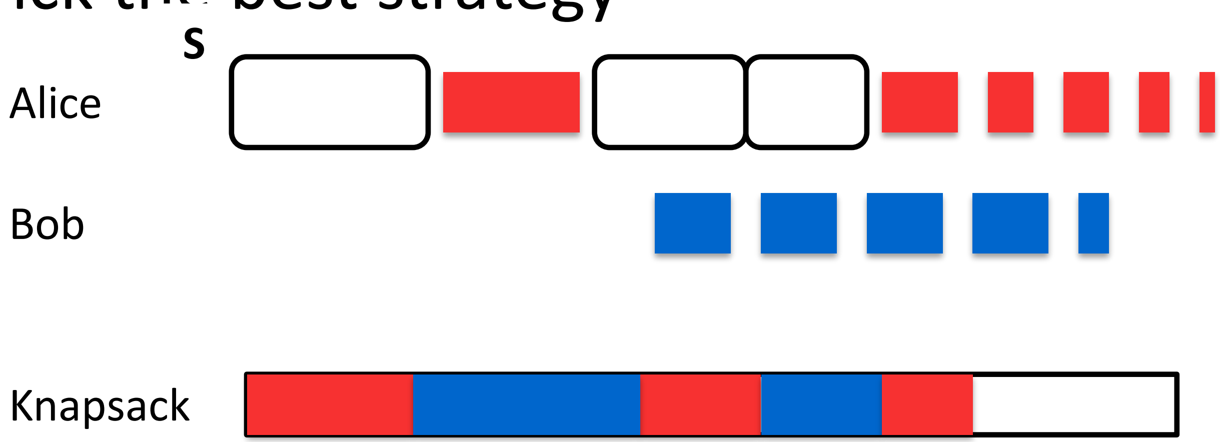
PTAS

- For each $S \subseteq$ Alice-items with $|S| \leq 1/\varepsilon$
  - Choose items of S large to small
  - Continue greedily
- If impossible to pack S
  - Discard
- Pick the best strategy



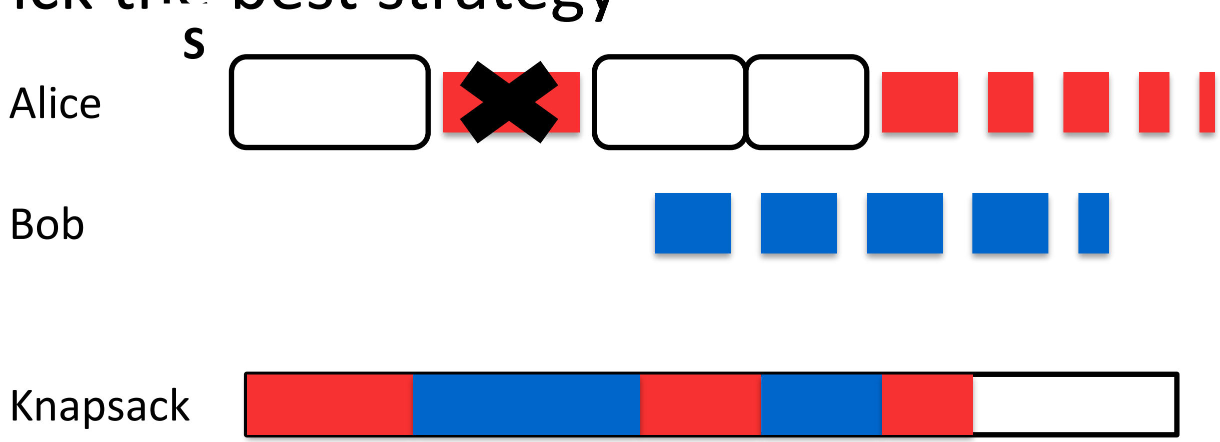
PTAS

- For each $S \subseteq$ Alice-items with $|S| \leq 1/\varepsilon$
  - Choose items of S large to small
  - Continue greedily
- If impossible to pack S
  - Discard
- Pick the best strategy



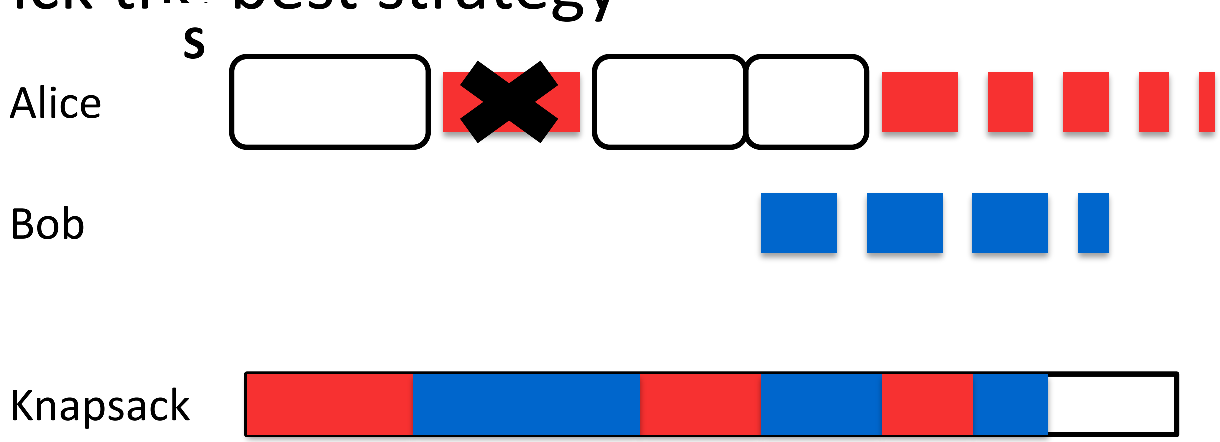
PTAS

- For each $S \subseteq$ Alice-items with $|S| \leq 1/\varepsilon$
  - Choose items of S large to small
  - Continue greedily
- If impossible to pack S
  - Discard
- Pick the best strategy



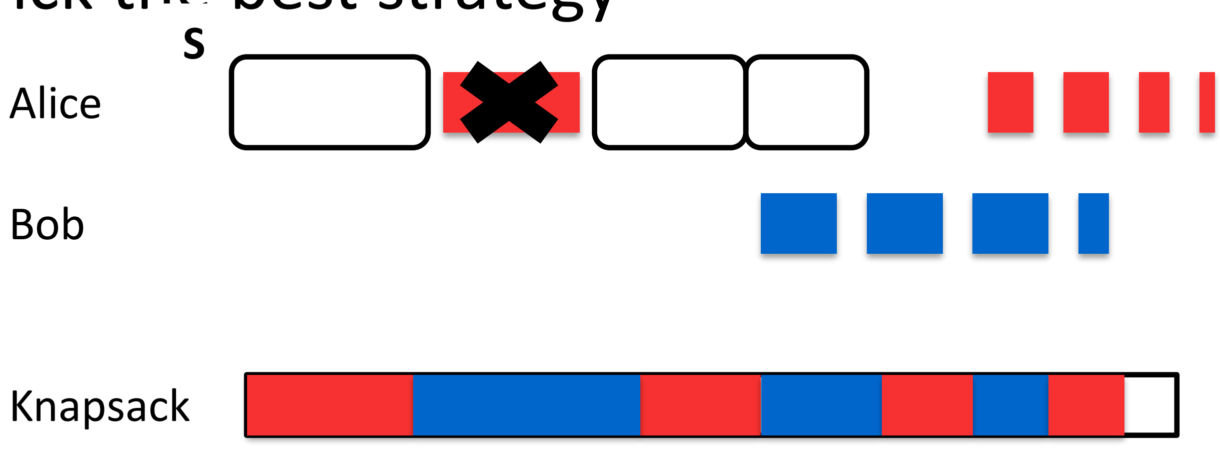
PTAS

- For each $S \subseteq$ Alice-items with $|S| \leq 1/\varepsilon$
  - Choose items of S large to small
  - Continue greedily
- If impossible to pack S
  - Discard
- Pick the best strategy



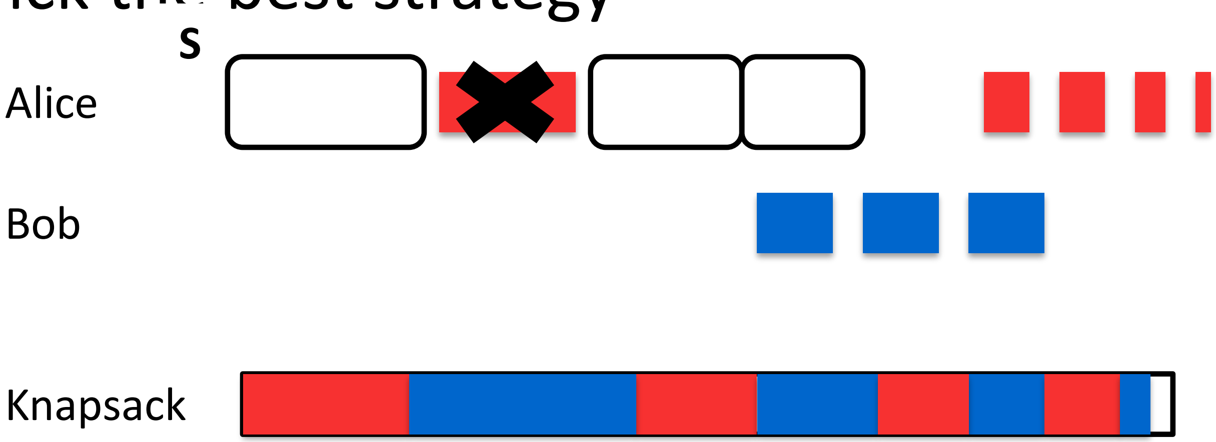
PTAS

- For each $S \subseteq$ Alice-items with $|S| \leq 1/\varepsilon$
  - Choose items of S large to small
  - Continue greedily
- If impossible to pack S
  - Discard
- Pick the best strategy



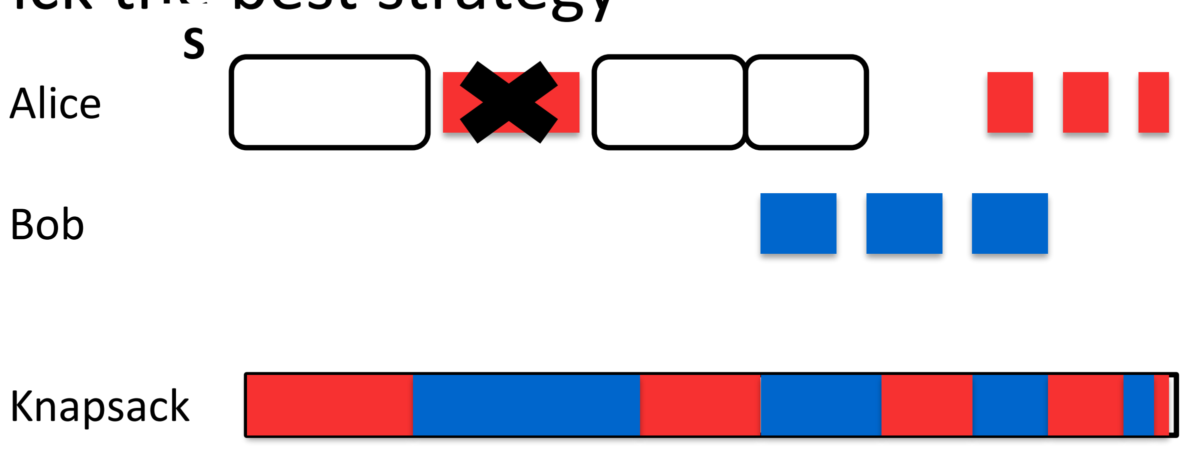
PTAS

- For each $S \subseteq$ Alice-items with $|S| \leq 1/\varepsilon$
  - Choose items of S large to small
  - Continue greedily
- If impossible to pack S
  - Discard
- Pick the best strategy



PTAS

- For each $S \subseteq$ Alice-items with $|S| \leq 1/\varepsilon$
  - Choose items of S large to small
  - Continue greedily
- If impossible to pack S
  - Discard
- Pick the best strategy



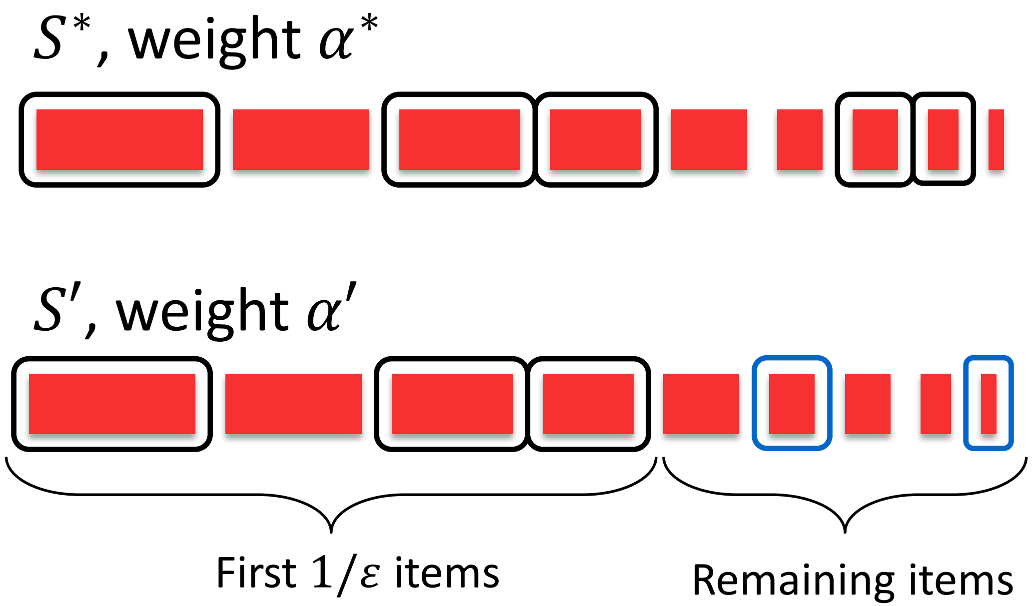
Approximation factor

S^* : optimal strategy

- Non-increasing order

S' :

- First $1/\varepsilon$ items of S^*
- Continue greedily



PTAS considered S'

- Finds a strategy of weight at least α'

Show $\alpha' \geq (1 - \varepsilon) \cdot \alpha^*$

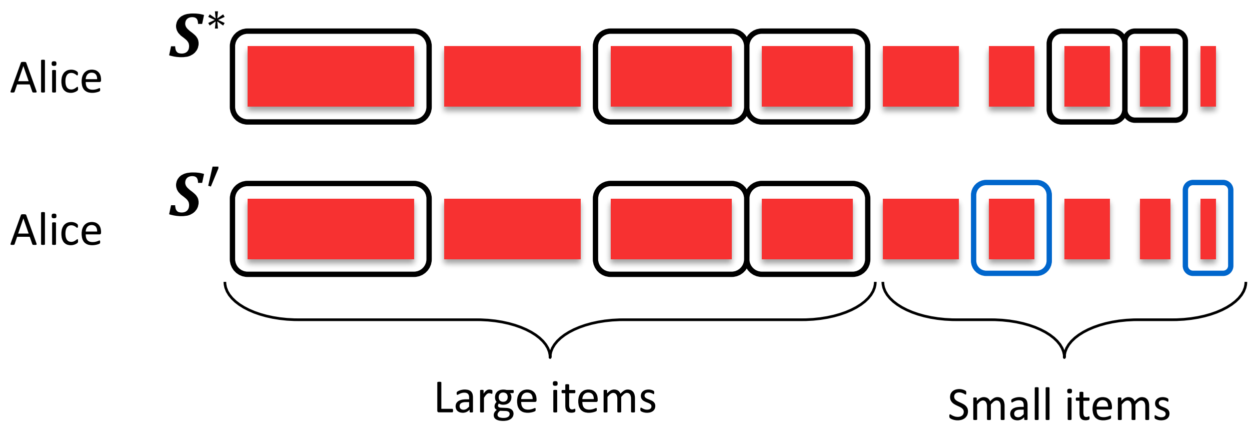
Approximation factor

Weight of the first $1/\varepsilon$ items

- Equal

Weight of the remaining items

- Remaining items are small: $\mathrm{a_i} \leq \varepsilon \cdot \alpha^*$
  - Else, first $1/\varepsilon$ items have weight at least α^*



Approximation factor

Difference: mistake by greedy when packing small

Lemma

Difference between greedy and selfish strategy is bounded by the size of the largest item of Alice (when playing against greedy Bob)

- Size of largest item $\leq \varepsilon\alpha^*$
- Thus $\alpha' \geq (1-\varepsilon)\alpha^*$

Running time

Try all size $1/\varepsilon$ subsets

- $O\left(n^{1/\varepsilon}\right)$

Continue greedily

- $O(n^2)$

Polynomial in n for ε constant

Theorem
The subset sum game against a greedy adversary has a PTAS

Note: not an FPTAS

No FPTAS

PTAS with running time polynomial in

- n
- $1/\varepsilon$

Theorem

The subset sum game against a greedy adversary has no FPTAS, unless P = NP

FPTAS gives polynomial-time algorithm for PARTITION

Inapproximability against the selfish player

Inapproximability

Theorem

A constant-factor approximation for the weight of Alice against selfish, implies $P = NP$.

Reduction from Partition (NP-hard)

- Input: Items $X = x_1, x_2, \ldots, x_n$ with $\sum x_i = 2\mathrm{U}$
- Question: Does there exist $S \subset X$ with sum U?
  - Partition X into 2 sets with equal sum

Proof strategy

Given instance for Partition

- Construct an instance for the Subset Sum game
- If yes-instance for Partition
  - Alice can pack at most n
- If no-instance for Partition
  - Alice can pack more than n/α

- Run α-approximation algorithm for Subset Sum game
  - Weight less than n, return yes
  - Weight more than n, return no

Reduction

Given $x_1, \dots, x_n$ for partition, with sum $2U$

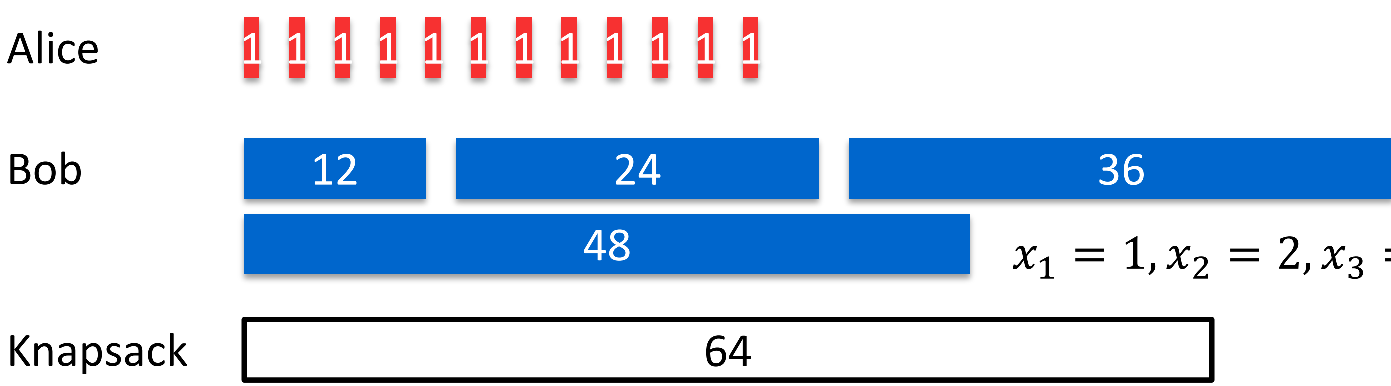
Give to Alice

- M items of weight 1

Give to Bob

- An item of weight $M * x_i$ for each $i \in [n]$

Let the capacity be $c = M * U + n$

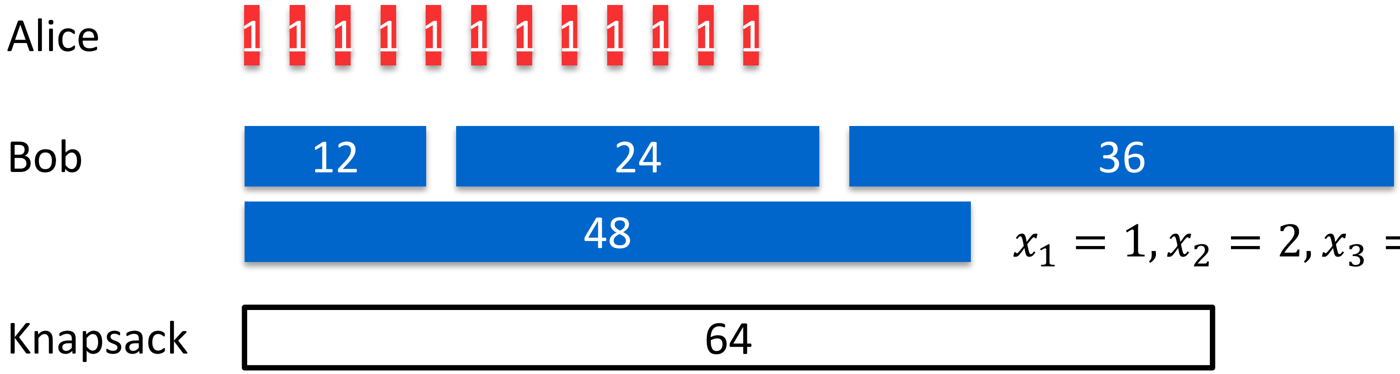


$x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 4$

$R = 3, n = 4, U = 5$

Reduction

- Partition was a yes-instance
  - Bob packs $M \cdot U = \mathrm{c} - n$
  - Alice can only pack weight n

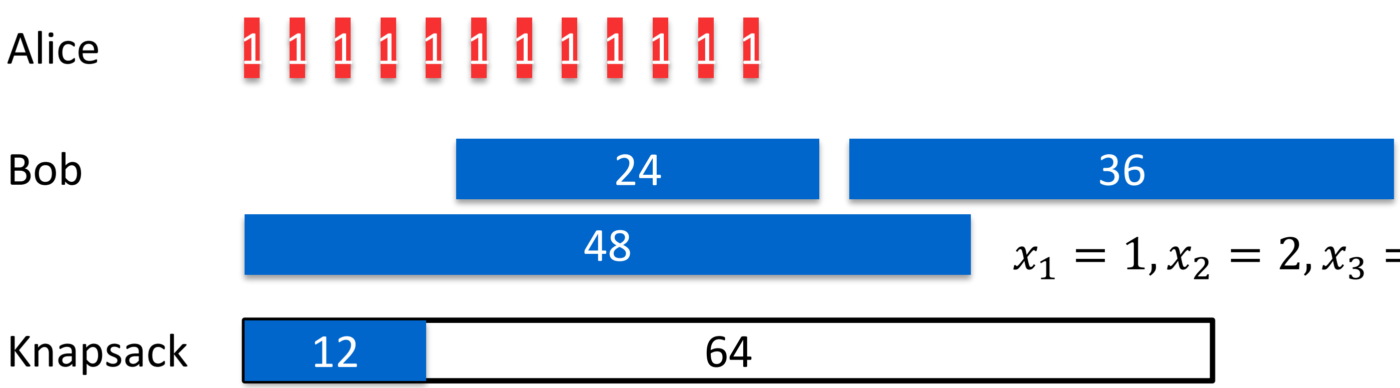


$x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 4$

$R = 3, n = 4, U = 5$

Reduction

- Partition was a yes-instance
  - Bob packs $M \cdot U = \mathrm{c} - n$
  - Alice can only pack weight n

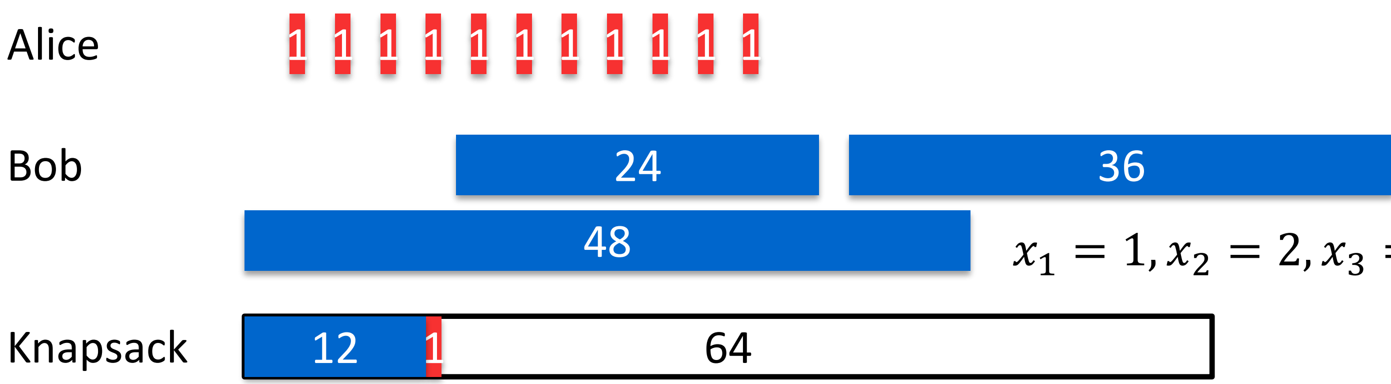


$x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 4$

$R = 3, n = 4, U = 5$

Reduction

- Partition was a yes-instance
  - Bob packs $M \cdot U = \mathrm{c} - n$
  - Alice can only pack weight n

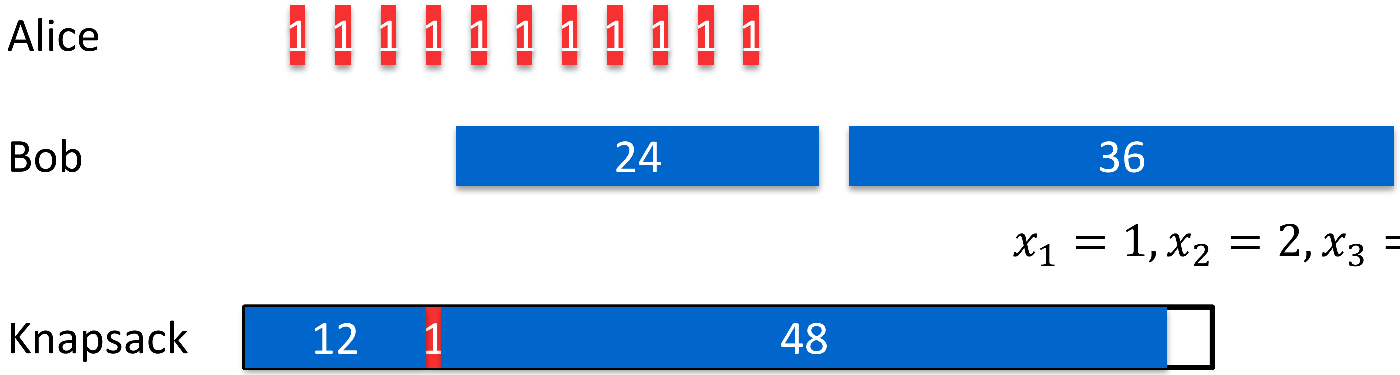


$x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 4$

$R = 3, n = 4, U = 5$

Reduction

- Partition was a yes-instance
  - Bob packs $M \cdot U = \mathrm{c} - n$
  - Alice can only pack weight n

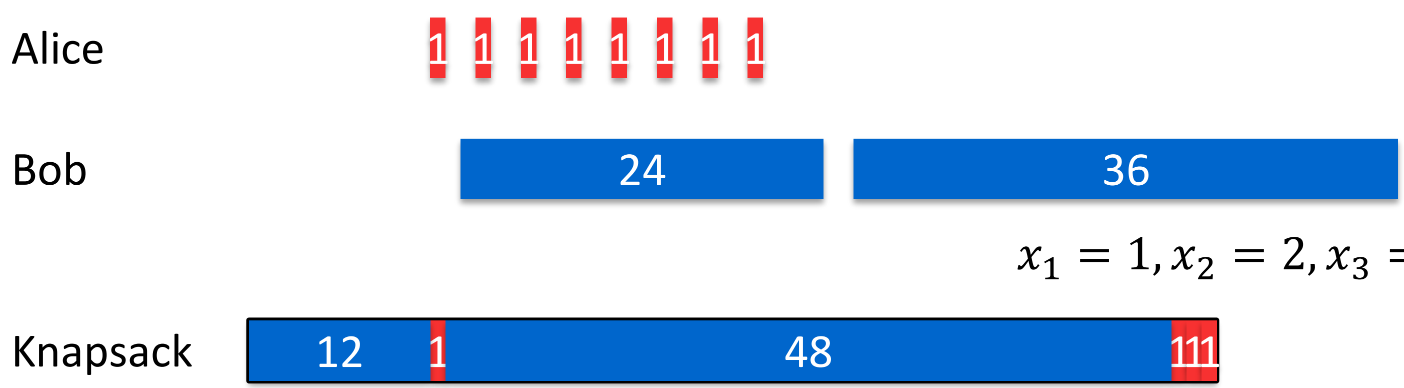


$x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 4$

$R = 3, n = 4, U = 5$

Reduction

- Partition was a yes-instance
  - Bob packs $M \cdot U = \mathrm{c} - n$
  - Alice can only pack weight n

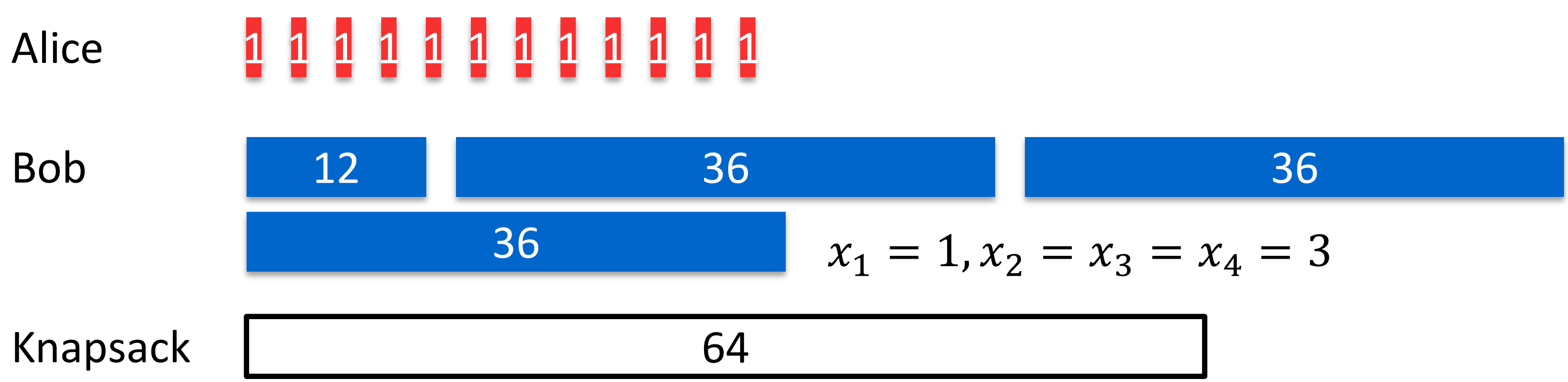


$x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 4$

$R = 3, n = 4, U = 5$

Reduction

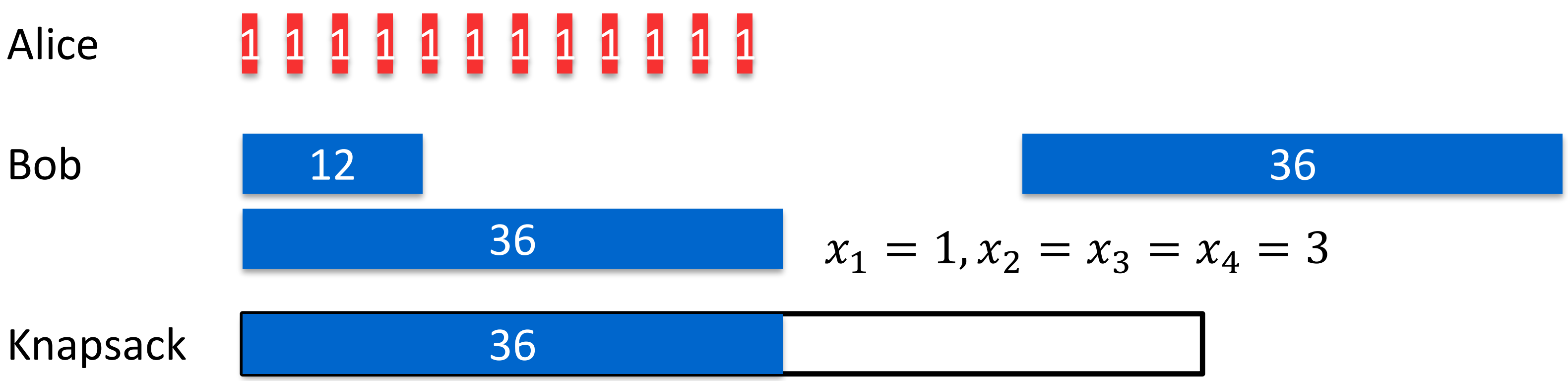
- Partition was a yes-instance
  - Bob packs $M \cdot U = \mathrm{c} - n$
  - Alice can only pack weight n
- Partition was a no-instance
  - Bob can pack at most $M(U-1)$
  - Alice can place at least weight M



$R = 3, n = 4, U = 5$

Reduction

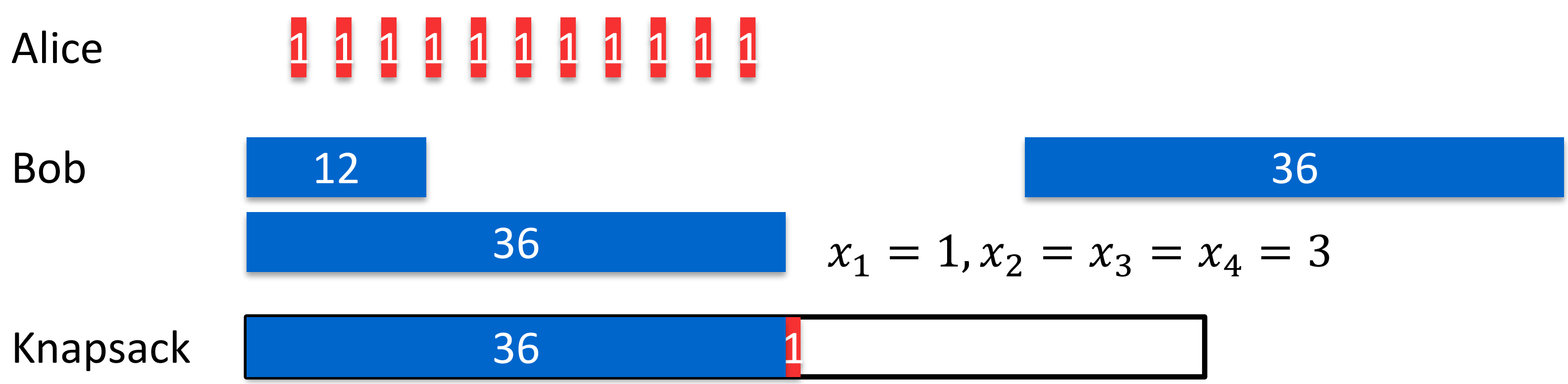
- Partition was a yes-instance
  - Bob packs $M \cdot U = \mathrm{c} - n$
  - Alice can only pack weight n
- Partition was a no-instance
  - Bob can pack at most $M(U-1)$
  - Alice can place at least weight M



$R = 3, n = 4, U = 5$

Reduction

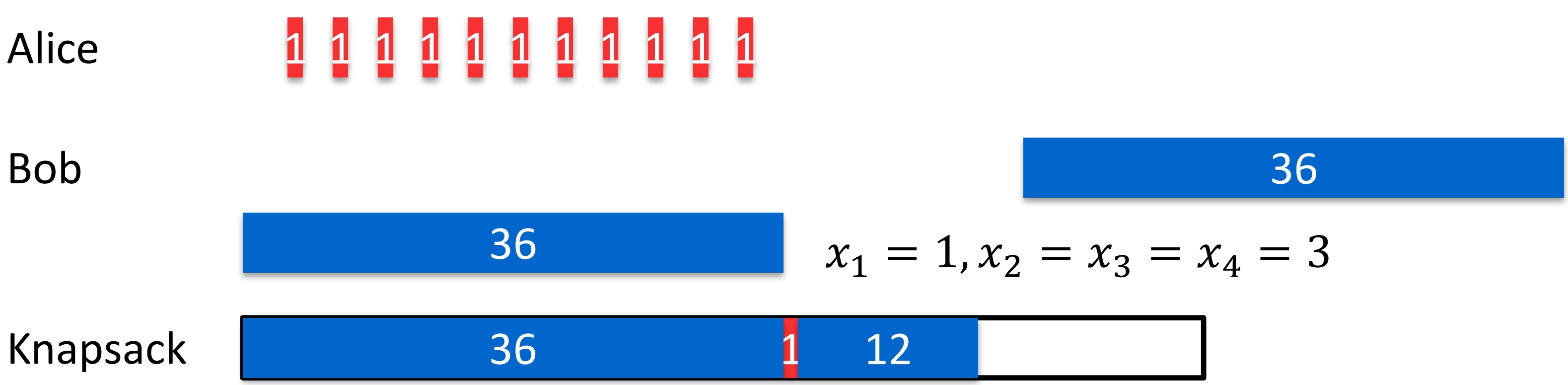
- Partition was a yes-instance
  - Bob packs $M \cdot U = \mathrm{c} - n$
  - Alice can only pack weight n
- Partition was a no-instance
  - Bob can pack at most $M(U-1)$
  - Alice can place at least weight M



$R = 3, n = 4, U = 5$

Reduction

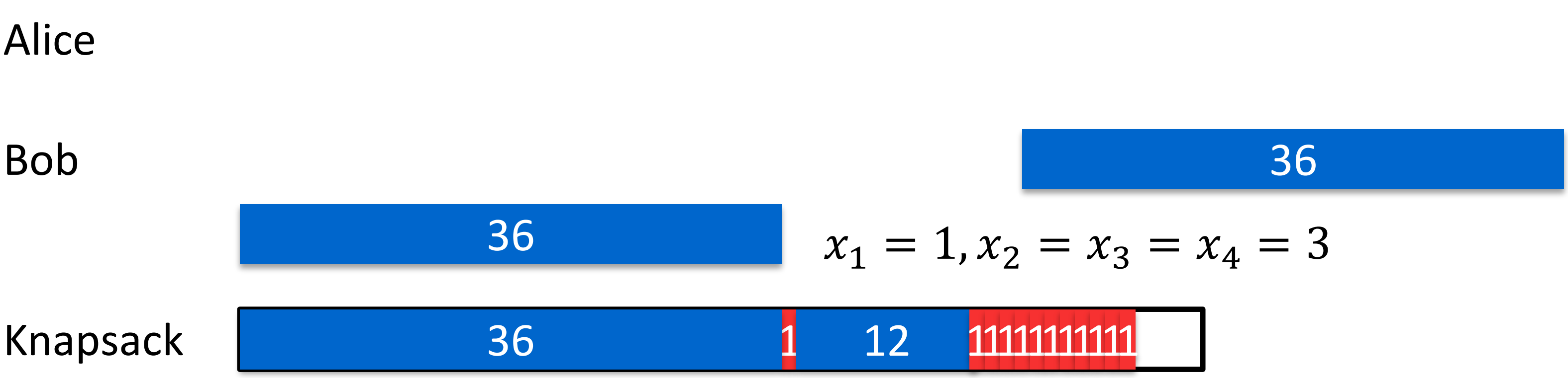
- Partition was a yes-instance
  - Bob packs $M \cdot U = \mathrm{c} - n$
  - Alice can only pack weight n
- Partition was a no-instance
  - Bob can pack at most $M(U-1)$
  - Alice can place at least weight M



$R = 3, n = 4, U = 5$

Reduction

- Partition was a yes-instance
  - Bob packs $M \cdot U = \mathrm{c} - n$
  - Alice can only pack weight n
- Partition was a no-instance
  - Bob can pack at most $M(U-1)$
  - Alice can place at least weight M



$R = 3, n = 4, U = 5$

Reduction

α-approximation for Subset Sum game

$\Downarrow$

polynomial-time algorithm for Partition

Theorem

A constant-factor approximation for the weight of Alice against selfish, implies $P = NP$.

The same holds against a hostile player

Pseudo-polynomial time algorithm

Algorithm against greedy

Theorem
The game against greedy is solvable in time $O(n^2 m^2 c^4)$

Use dynamic programming

$[i, j, W_A, W_B] :=$ Maximum weight Alice can obtain

- It is Alice's turn
- Weight of Alice is W_A, weight of Bob W_B
- Alice just packed i, Bob j

Take state with best-reachable W_A

Reachability

Check whether a state is reachable from another

$[i, j, W_A, W_B]$ to $[i', j', W_A', W_B']$

- $i < i'$
- $j < j'$
- $W_A' = W_A + a_{i'}$
- $W_B' = W_B + b_{j'}$
- The items still fit
- $b_{j'}$ was the largest available Bob-item
  - Thus the Greedy choice

Conclusion

SSG against hostile/selfish

- $PSPACE$ complete
- No α-approximation unless $P = NP$

SSG against greedy

- Solvable in pseudo polynomial time
- Has a PTAS
- Has no FPTAS unless $P = NP$