Brief journey in the big world of Integer Programming with the Knapsack

Olivier Pirson

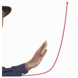
Presentation for the oral exam of
INFO-F424 *Combinatorial optimization*

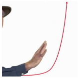
Computer Science Department
Université Libre de Bruxelles

ULB

June 1st, 2017
(Cosmetic correction November 26, 2017)

Last version: `https://bitbucket.org/OPiMedia/brief-journey-in-the-big-world-of-integer-programming-with-the/`





$0 - 1$ Knapsack Problem formulation

Let

- a set of n items, for each $i \in \{1, 2, \ldots, n\}$
  - a weight a_i
  - a value c_i
- a knapsack of capacity b

Find the subset of $\{1, 2, \ldots, n\}$
that it may be contained in the knapsack *and* maximizes the value.

The canonical Binary Integer Program formulation:

Variables: $x_i \in \{0, 1\}, x_i = \begin{cases} 1 & \text{if item } i \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$

Constraint: $\sum_{i=1}^{n} a_i x_i \leq b$

Objective function: $\max \sum_{i=1}^{n} c_i x_i$

Greedy heuristic algorithm – primal/lower bound

Sort variables by decreasing order of density $\frac{\text{value}}{\text{weight}}$: $\frac{c_i}{a_i}$

In this order, pick each possible item.
That fill the knapsack with the best *independent* choice.

For example:

max	$45x_1$	$+ 48x_2$	$+ 9x_3$	
subject to	$5x_1$	$+ 12x_2$	$+ 3x_3$	≤ 16
density	9	4	3	

Give the solution: $(1, 0, 1)$ with value $45 + 9 = 54$.

54 is a lower bound of the problem.

Indeed $(0, 1, 1)$ is a better solution (in fact the optimal solution [1]) of value $48 + 9 = 57$.

[1] In simple example like this
it is very easy to perform an exhaustive search.

Linear relaxation – dual/upper bound

The idea is to permit piece of item, and then fill completely the knapsack. Of course the relaxed problem is an other problem and its solution is not a solution of the initial problem.
But this solution is an upper bound for the initial problem.

With the same example:

$$\begin{array}{ll} \text{max} & 45x_1 + 48x_2 + 9x_3 \\ \text{subject to} & 5x_1 + 12x_2 + 3x_3 \leq 16 \end{array}$$

and now $x_i \in [0, 1]$ instead $\in \{0, 1\}$

$5 + \alpha 12 = 16 \Longrightarrow \alpha = \frac{11}{12}$
Give $(1, \frac{11}{12}, 0)$ with value $45 + \frac{11}{12}48 = 89$.

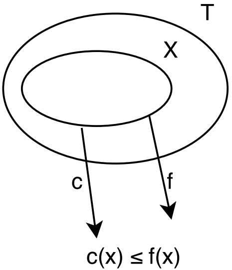
We known now that the optimal solution $x^\star$ of the initial problem is such that: $54 \leq x^\star \leq 89$

We obtained theses two bound in $O(n)$.

For the Integer Programming
$z = \max\{c(x) \mid x \in X \subseteq R^n\}$

$z^R = \max\{f(x) \mid x \in T \subseteq R^n\}$
is a relaxation if

- $X \subseteq T$
- $c(x) \leq f(x) \qquad \forall x \in X$



So z^R is an upper bound for the initial problem.

We can break the problem into two subproblems, and so forth.
That give this kind of binary tree representation:

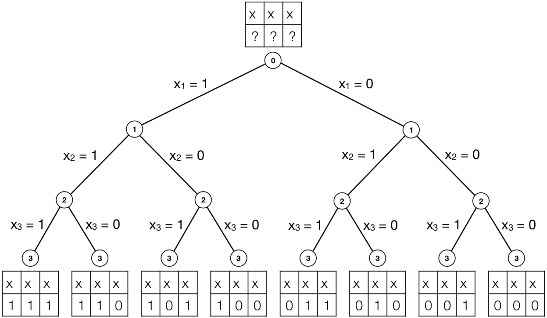


Figure: ***Discrete Optimization***

Contain the Exponential Explosion

There is 2^n possibilities, so an exhaustive search is quickly impossible.

In fact, it is a NP-hard problem.
(The decision problem is already NP-complete.)

The goal is to push as far as possible the exponential curve.

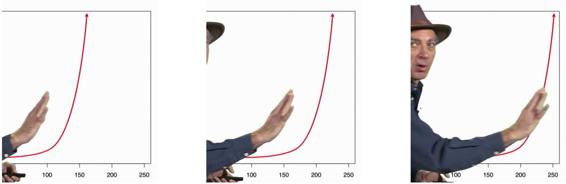
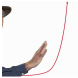


Figure: ***Discrete Optimization***

How avoid the impossible (in practice) exhaustive search?



Pruning. . .

i	V	W
1	45	5
2	48	8
3	35	3

K = 10

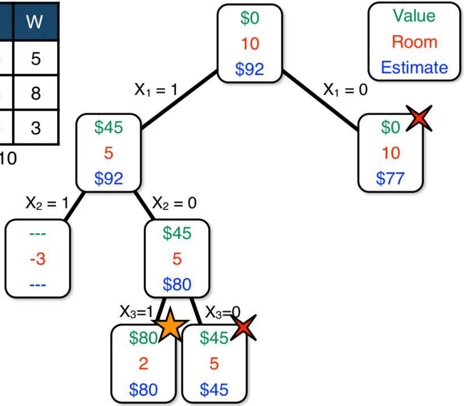
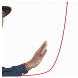


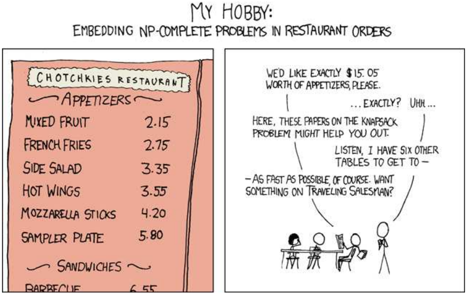
Figure: ***Discrete Optimization***

We have the optimal solution 80 with only 7 nodes visited, instead $2^4 - 1 = 15$.



References:

- Pascal Van Hentenryck.
  Discrete Optimization.
  MOOC, University of Melbourne,
  https://www.coursera.org/learn/discrete-optimization
- Laurence A. Wolsey.
  Integer Programming.
  Wiley, 1998



***NP-Complete*, xkcd**