Knapsack and Important digressions

we will see a variety of important CS concepts. $\Rightarrow$ 11/30.

Input n items, each associated with a value and a size/weight $\{(v_i, w_i)\}_{i=1..n}$

– Knapsack capacity. B.

Output

– $S \subseteq [n]$ such that the total value $\sum_{i \in S} v_i$ is maximized subject to the capacity constraint $\sum_{i \in S} w_i \le B$.

Integer Programming formulation.

An IP problem is a mathematical optimization.

$$\begin{aligned} \max_{x \in \mathbb{Z}^n} \quad & c^T x \\ \boxed{\text{s.t.}} \quad & Ax \le b \\ & x \ge 0 \end{aligned}$$

Integer linear program

Knapsack as an ILP.

$$\max_{x_1,\dots,x_n} \sum_{i=1}^{n} x_i v_i .$$

$$\boxed{\text{s.t.}} \quad \sum x_i w_i \le B$$

$$x_i \in \{0,1\} \quad i=1..n.$$

Digression: Let's write another combinatorial optimization problem as an ILP.

$$\min \sum_{i=1}^{n} x_i$$

$$x_i + x_j \geq 1 \quad \forall (i,j) \in E.$$

$$x_i \in \{0,1\}.$$

$\Rightarrow$ VERTEX COVER problem.

Consider the decision form "Can a value of (at least) V be achieved (i.e., without exceeding the weight W)?

$\Rightarrow$ Claim $\Downarrow$ NP-complete.

Linear Programming relaxation.

$$\left.\begin{aligned} \max \quad & c^T x. \\ \text{s.t} \quad & Ax \leq b. \\ & x \geq 0. \end{aligned}\right]$$ linear program standard form.

Example

max $\quad x_1 + x_2$

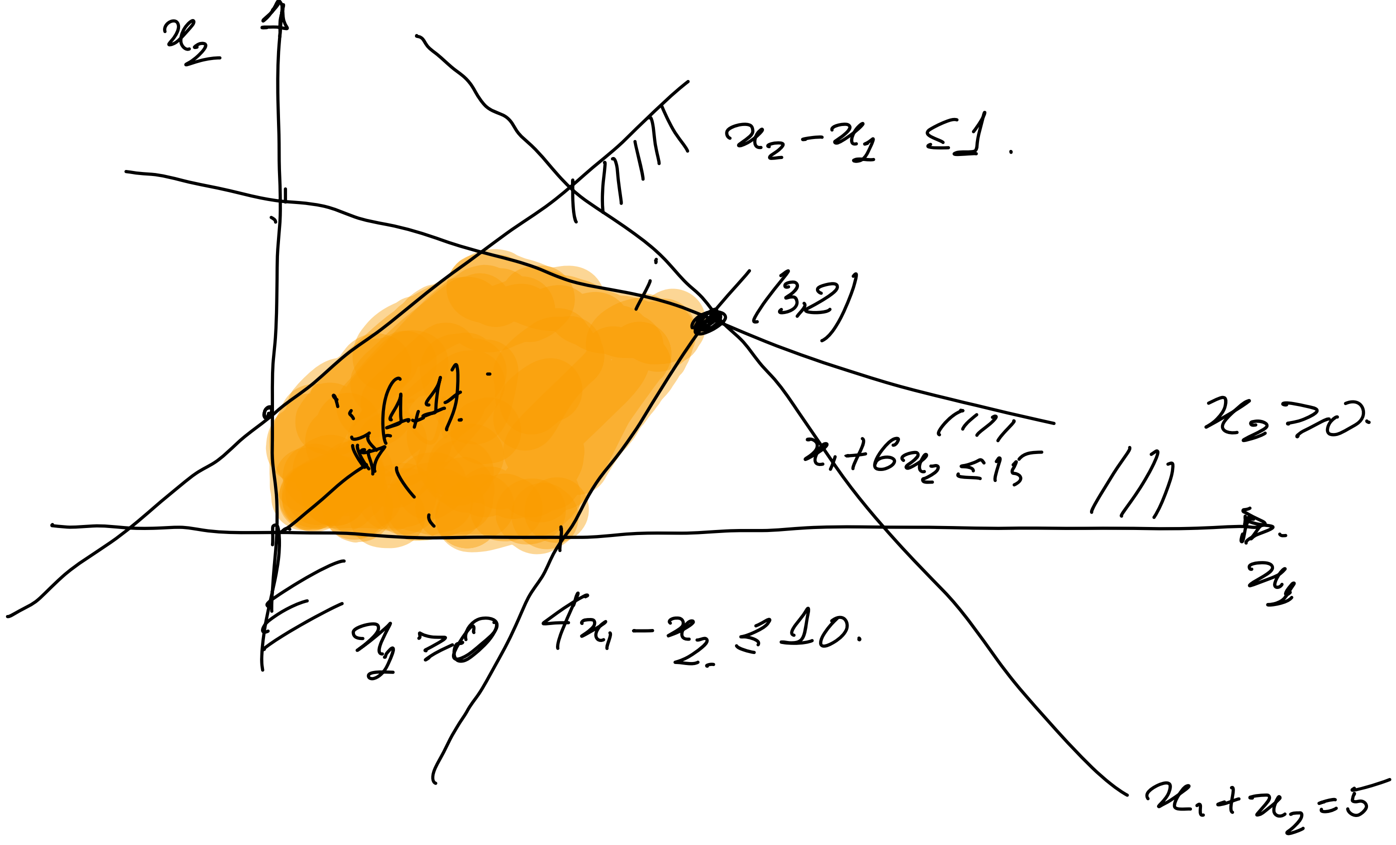
subject to. $\quad x_1 \geq 0$

$x_2 \geq 0.$

$x_2 - x_1 \leq 1$

$x_1 + 6x_2 \leq 15$

$4x_1 - x_2 \leq 10.$



A LP is the problem of maximizing a linear function over the set of all vectors that satisfy a given system of linear eqs. and linear inequalities. Each LP can be transformed to.

$$\max \; c^T x.$$
$$\text{subject to. } Ax \leq b.$$

Examples from Matoušek, Gärtner.

① Diet

The Office of Nutrition Inspection of the EU recently found out that dishes served at the dining and beverage facility "Bullneck's," such as herring, hot dogs, and house-style hamburgers do not comport with the new nutritional regulations, and its report mentioned explicitly the lack of vitamins A and C and dietary fiber. The owner and operator of the aforementioned facility is attempting to rectify these shortcomings by augmenting the menu with vegetable side dishes, which he intends to create from white cabbage, carrots, and a stockpile of pickled cucumbers discovered in the cellar. The following table summarizes the numerical data: the prescribed amount of the vitamins and fiber per dish, their content in the foods, and the unit prices of the foods.[1]

Food	Carrot, Raw	White Cabbage, Raw	Cucumber, Pickled	Required per dish
Vitamin A [mg/kg]	35	0.5	0.5	0.5 mg
Vitamin C [mg/kg]	60	300	10	15 mg
Dietary Fiber [g/kg]	30	20	10	4 g
price [€ /kg]	0.75	0.5	0.15*	—

*Residual accounting price of the inventory, most likely unsaleable.

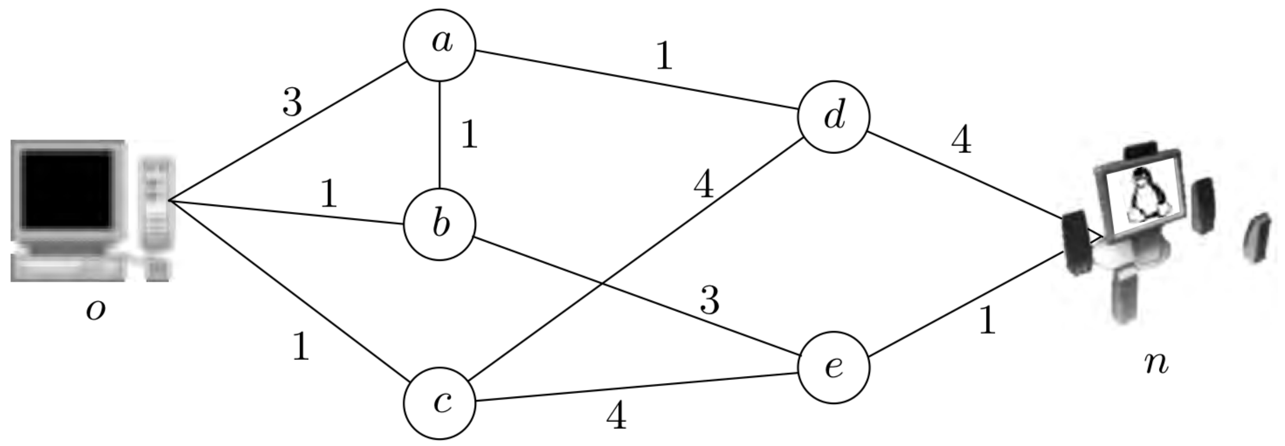
At what minimum additional price per dish can the requirements of the Office of Nutrition Inspection be satisfied? This question can be expressed by the following linear program:

$$\begin{array}{ll}
\text{Minimize} & 0.75x_1 + 0.5x_2 + 0.15x_3 \\
\text{subject to} & x_1 \geq 0 \\
& x_2 \geq 0 \\
& x_3 \geq 0 \\
& 35x_1 + 0.5x_2 + 0.5x_3 \geq 0.5 \\
& 60x_1 + 300x_2 + 10x_3 \geq 15 \\
& 30x_1 + 20x_2 + 10x_3 \geq 4.
\end{array}$$

$\textcircled{2}$ Max-flow as LP

2.2 Flow in a Network

An administrator of a computer network convinced his employer to purchase a new computer with an improved sound system. He wants to transfer his music collection from an old computer to the new one, using a local network. The network looks like this:



What is the maximum transfer rate from computer o (old) to computer n (new)? The numbers near each data link specify the maximum transfer rate of that link (in Mbit/s, say). We assume that each link can transfer data in either direction, but not in both directions simultaneously. So, for example, through the link ab one can *either* send data from a to b at any rate from 0 up to 1 Mbit/s, *or* send data from b to a at any rate from 0 to 1 Mbit/s.

The nodes $a, b, \ldots, e$ are not suitable for storing substantial amounts of data, and hence all data entering them has to be sent further immediately. From this we can already see that the maximum transfer rate cannot be used on all links simultaneously (consider node a, for example). Thus we have to find an appropriate value of the data flow for each link so that the total transfer rate from o to n is maximum.

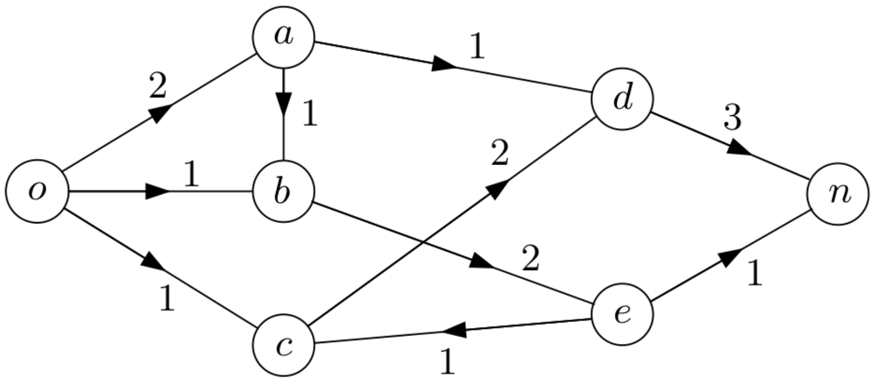
For every link in the network we introduce one variable. For example, x_{be} specifies the rate by which data is transfered from b to e. Here x_{be} can also be negative, which means that data flow in the opposite direction, from e to b. (And we thus do *not* introduce another variable x_{eb}, which would correspond to the transfer rate from e to b.) There are 10 variables: x_{oa}, x_{ob}, x_{oc}, x_{ab}, x_{ad}, x_{be}, x_{cd}, x_{ce}, x_{dn}, and x_{en}.

We set up the following linear program:

$$
\begin{array}{ll}
\text{Maximize} & x_{oa} + x_{ob} + x_{oc} \\
\text{subject to} & -3 \le x_{oa} \le 3, \quad -1 \le x_{ob} \le 1, \quad -1 \le x_{oc} \le 1 \\
& -1 \le x_{ab} \le 1, \quad -1 \le x_{ad} \le 1, \quad -3 \le x_{be} \le 3 \\
& -4 \le x_{cd} \le 4, \quad -4 \le x_{ce} \le 4, \quad -4 \le x_{dn} \le 4 \\
& -1 \le x_{en} \le 1 \\
& x_{oa} = x_{ab} + x_{ad} \\
& x_{ob} + x_{ab} = x_{be} \\
& x_{oc} = x_{cd} + x_{ce} \\
& x_{ad} + x_{cd} = x_{dn} \\
& x_{be} + x_{ce} = x_{en}.
\end{array}
$$

The objective function $x_{oa} + x_{ob} + x_{oc}$ expresses the total rate by which data is sent out from computer o. Since it is neither stored nor lost (hopefully) anywhere, it has to be received at n at the same rate. The next 10 constraints, $-3 \le x_{oa} \le 3$ through $-1 \le x_{en} \le 1$, restrict the transfer rates along the individual links. The remaining constraints say that whatever enters each of the nodes a through e has to leave immediately.

The optimal solution of this linear program is depicted below:



The number near each link is the transfer rate on that link, and the arrow determines the direction of the data flow. Note that between c and e data has to be sent in the direction from e to c, and hence $x_{ce} = -1$. The optimum value of the objective function is 4, and this is the desired maximum transfer rate.

· LP AS A TOOL of designing APPROXIMATION Algorithms.

Vertex cover (VC).

IP $\left\{ \begin{array}{lll} \min & \sum_{v \in V} x_v . & \\ \text{s.t.} & x_u + x_v \geq 1 & \forall \text{edge } (u,v) \\ & x_v \in \{0,1\} & \forall v \in V \end{array} \right.$

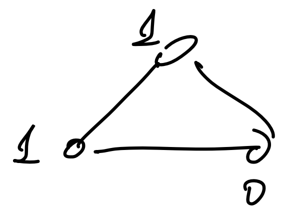
VC is NP-HARD. We relax the IP to the following LP.

Solvable in poly-time!

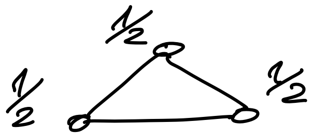
$$\begin{array}{lll} \min & \sum_{v \in V} x_v & \\ \text{s.t.} & x_u + x_v \geq 1 & \forall \text{edge } (u,v) \\ & 0 \leq x_v \leq 1 & \forall v \in V . \end{array}$$

Clearly $OPT_{LP} \leq OPT_{IP}$ (why?).

However we are no longer dealing with an indicator optimal solution.



vs



Let x^*_{LP} be the optimal $[0,1]^n$ solution of ~~the~~ LP

$$\Big\Downarrow \text{ rounding.}$$

binary $\tilde{x}$

Here we let $\tilde{x}_v = 1$ only if $x^{LP}_v \geq 1/2$.

Let $S_{LP} = \{ v \in V : x^{*LP}_v \geq 1/2 \}$.

Theorem $|S_{LP}| \leq 2\ OPT_{IP}$.

Proof

$$|S_{LP}| = \sum_{v : x^*_v \geq 1/2} 1 \leq \sum_{v \in S_{LP}} 2x^*_v \leq \sum_{v \in V} 2x^*_v$$

$$= 2\ OPT_{LP} \leq 2\ OPT_{IP}.$$

$\boxtimes$

BACK to the KNAPSACK

The knapsack problem can be solved in pseudopolynomial time $O(nB)$ ($n = \#$items, B knapsack capacity).

Dfn Pseudopolynomial means polynomial in the inputsize if the input is represented in unary.

(it would have been polynomial if the complexity were $O(n \log_2 B)$ but this is very unlikely unless $P=NP$).

Question Does the greedy algorithm work (AS AN APPROXIMATION)? $\xrightarrow{}$ yes... AND $\xrightarrow{}$ no... } depends. what greedy Algorithm but

first consider sorting items by the ratio $\frac{\text{VAl}}{\text{weight}}$.

$(v_1 = 1.01,\ w_1 = 1)$ $(v_2 = 10^6,\ w = 10^6)$ $B = 10^6$.

We'd choose even if OPT $= 10^6$ (ARBITRARily BAd)

$\boxed{\text{Exercise}}$ Run greedy, let S_1 be output.

Let $S_2 = \{$item of largest ~~VAl~~ value$\}$.

Return whichever of S_1, S_2 has more value.

Prove this is $\frac{1}{2}$ APPROXIMAtion.