9. Modelling integer programs (i.P.)

Q: What are i.P.?

$$\min c^T x \quad \left.\right\} \text{ L.P.}$$
$$Ax = b$$
$$x \geq 0$$
$$x^T = (\xi_1, \xi_2 \dots \xi_m) \quad \left.\right\} \text{ integrality constraint}$$
$$\xi_i - \text{integer}$$

Q: Why are i.P. different from LP?

Short answer:

Optimality conditions:

- primal feasibility
- dual feasibility
- complementary slackness

... do not hold

Integer programs (c'ed)

$$\begin{cases} \max \xi_1 + \xi_2 \\ 5\xi_1 + 4\xi_2 \le 20 \quad | \cdot \bar{\pi}_1 \\ 2\xi_1 + 3\xi_2 \le 12 \quad | \cdot \bar{\pi}_2 \\ \xi_i \ge 0 \end{cases}$$

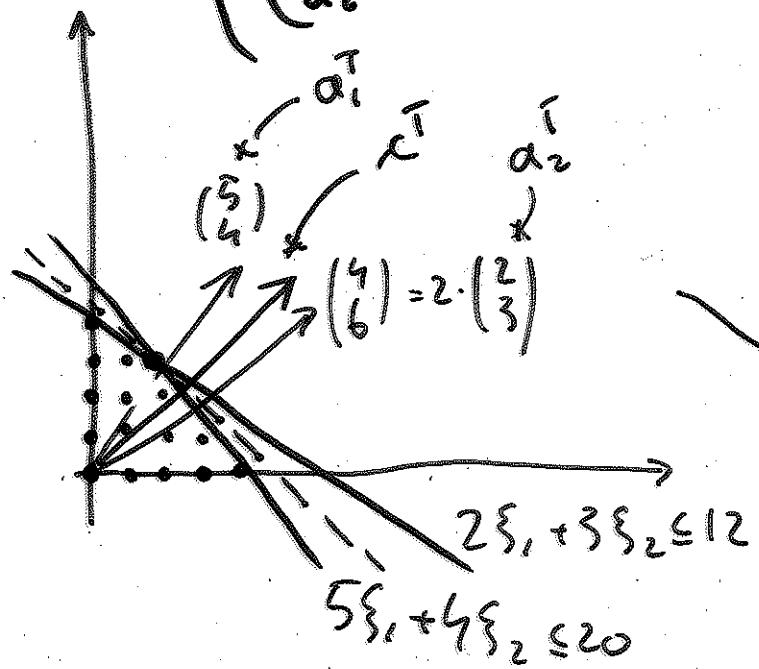
Primal

$$\begin{cases} \min 20\bar{\pi}_1 + 12\bar{\pi}_2 \\ \bar{\pi}_1 \begin{pmatrix} 5 \\ 4 \end{pmatrix} + \bar{\pi}_2 \begin{pmatrix} 2 \\ 3 \end{pmatrix} \ge \\ \ge \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\ \bar{\pi}_i \ge 0 \end{cases}$$

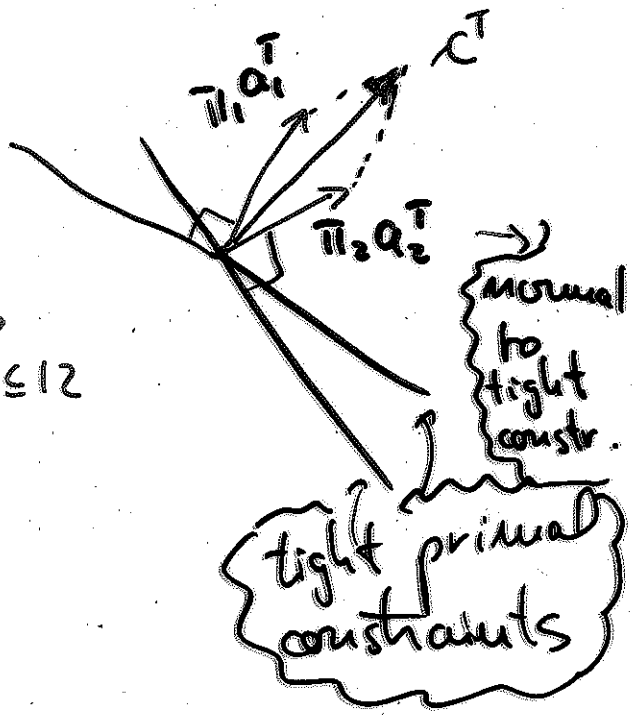
Dual

$$\Downarrow$$

$$\begin{cases} \max c^T x \\ \begin{pmatrix} a_1^T \\ a_2^T \end{pmatrix} x \le b \end{cases}$$

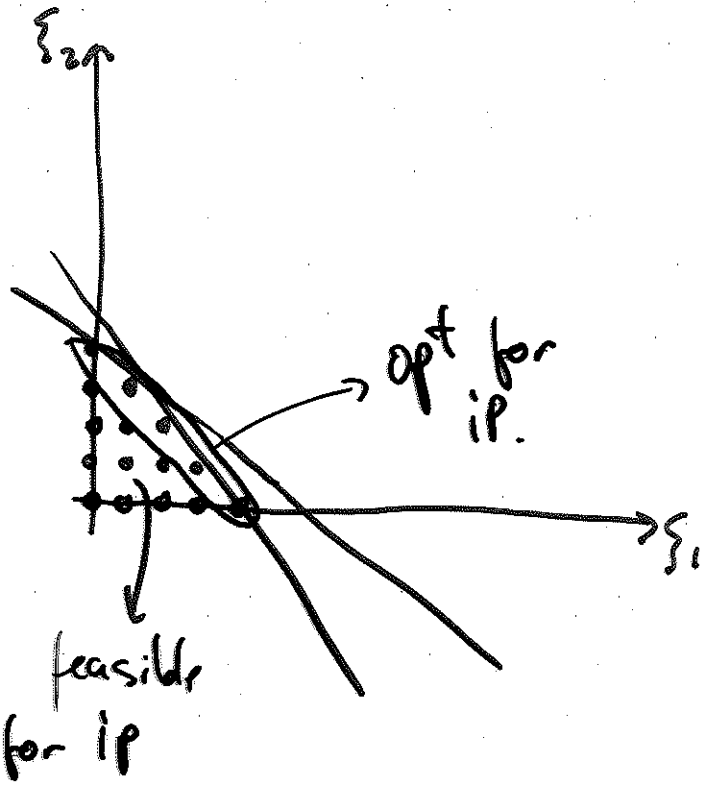


C.S. conditions & dual feasibility:



Integer Programs (c'ed)

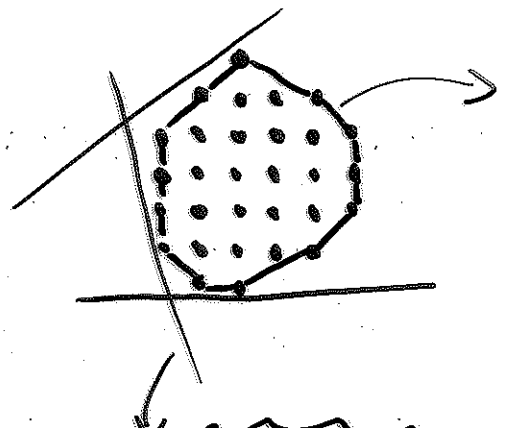
$$\max \xi_1 + \xi_2$$
$$5\xi_1 + 4\xi_2 \leq 20$$
$$2\xi_1 + 3\xi_2 \leq 12$$
$$\xi_1, \xi_2 \in \mathbb{Z}_{+0}$$



We do not have optimality conditions for IP whose set of feasible solutions is discrete

⋮

except if we compute the CONVEX HULL of feasible points



There is an exponential number of constraints to describe the convex hull of feasible I.P. points.

{original constraints of the problem

i.P. (c'od)

Problem complexity (basic notions)

Motivation: Certain reasonable problems turned out to be extremely difficult to solve with a computer.

Complexity Theory: to catalogue these reasonable problems in difficult, easy, etc... (the catalogue is A LOT MORE complex than that).

Q: What is a reasonable problem?

A1: Example of an ill-reasonable problem:

"What is the meaning of the Universe?"
(Douglas Adams)

A2: Another example of ill-reasonable pb:

"Enumerate all integers".

(A1) - answer cannot be checked.
example: 42 (Douglas Adams)

(A2) - answer is too long.

IP (c'ed)

Complexity (c'ed)

A3: decision problems (YES / NO answers) whose YES answers can be verified quickly (in polynomial time) if given a certificate.

EXAMPLE: Colouring problem: given a graph $G=(V,E)$ and an integer $k \in \mathbb{Z}_+$ does there $\exists$ a colouring of vertices with no more than k colours?

Certificate: a vertex colouring.

($O(n+m)$ to verify it is a valid k colouring

Obs Problems with a certificate for both YES & NO answers are more elusive... They are either "easy" problems ("Is the distance between A & B $\leq k$?) or a problem like factoring (given $N \in \mathbb{Z}$, and $k \in \mathbb{Z}$, does N have a factor $\leq k$?)

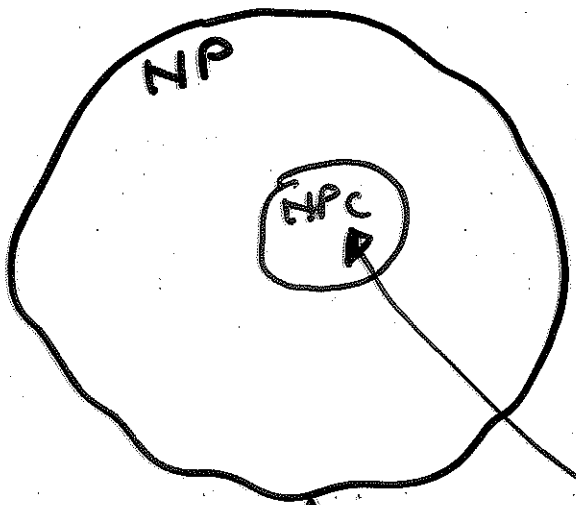
Complexity (c'ed)

def The "reasonable" problems defined earlier are called problems in class NP.

technical term: problems "solved" by a Non-deterministic Turing machine in Polynomial time.

Class NP-C (NP-complete problems)

NPC = the "core", the hard problems:



If someone finds a fast algorithm for ONE problem in NPC, then ALL problems in NP can be solved efficiently.

iP (c'ed)

Integer Programs are related to the NPC problems; they are at least as difficult as NPC (~~Theory~~ iP are called NP-hard probl.).

iP

$$\begin{cases} \min c^T x \\ Ax = b \\ x \in \mathbb{Z}_+^n \end{cases}$$

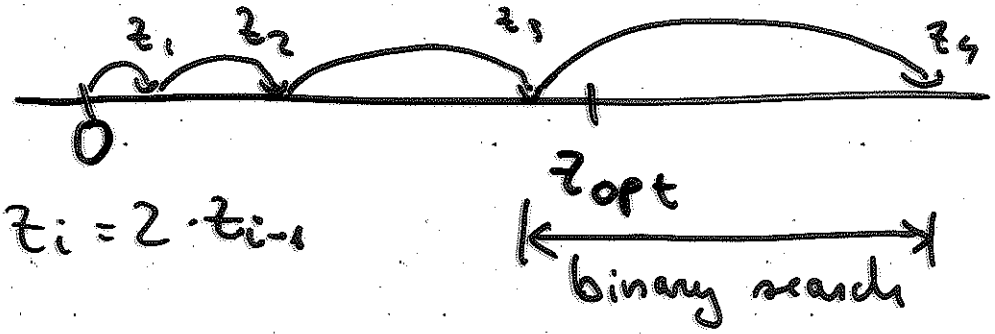
not in NP

decision-iP

given $z \in \mathbb{R}$, is there $x \in \mathbb{Z}_+^n$ so that $Ax = b$ and $c^T x \leq z$?

in NPC.

However, we can solve iP using calls to decision-iP in binary search (actually unbounded search) fashion



IP (cool)

Conclusion IP = NP-hard ; we cannot hope to use theoretically efficient algorithms to solve them.

Approach

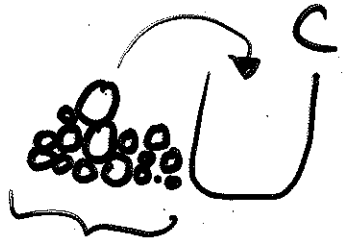
- study algorithms efficient in practice
- impose a time threshold T (eg: 2 h)
- run algorithm on IP
  - answer within T (IP solved optimally)
  - no answer within T (abandon IP).
- goal: solve IP with many variables & constraints.

In contrast: LP = easy theoretically. However, simplex does not have a good (polynomial time) performance in the worst case, but performs well in practice.

IP (c'ed)

Solving problems with I.P.

0-1 IP Knapsack.



I : articles

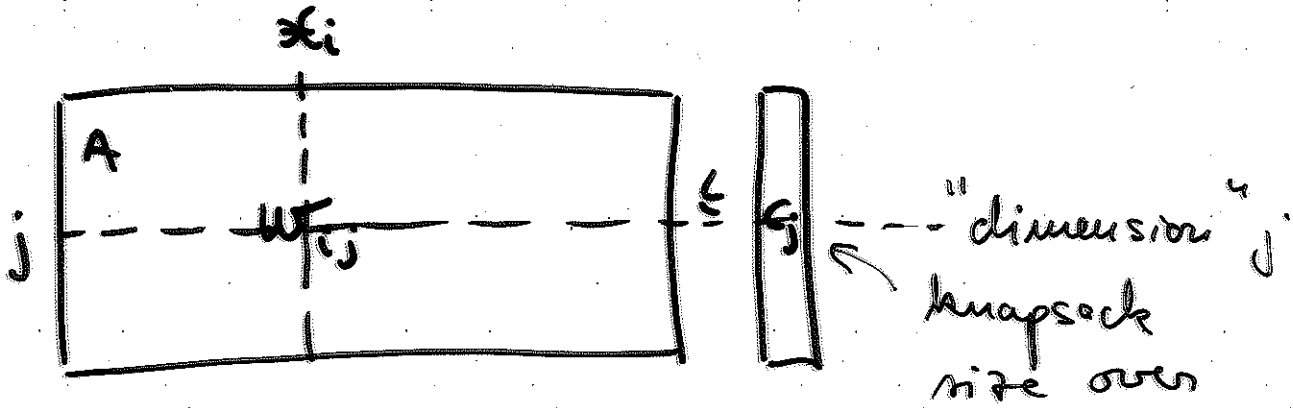
$i \in I$ - v_i = value

- w_i = weight

$$\begin{cases} \max \sum_{i \in I} v_i \cdot x_i \\ \sum_{i \in I} w_i x_i \leq C \\ x_i \in \{0, 1\} \end{cases}$$

$$x_i = \begin{cases} 0 : i \text{ not in Knapsack} \\ 1 : i \text{ in knapsack} \end{cases}$$

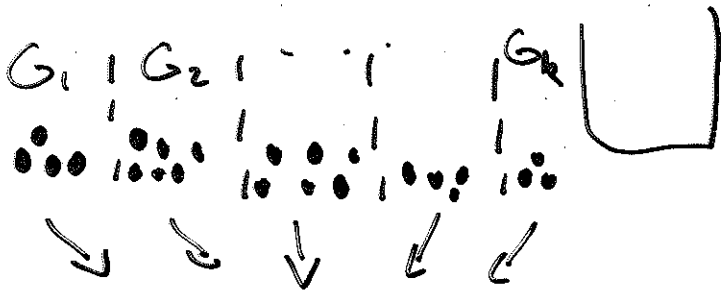
Multi-dimensional knapsack :



(the rest is same as for knapsack)

IP (cont'd)

Multi-dimensional Knapsack with Multiple-choice constraints



Multiple choice: Exactly one item selected from each group.

+

previous constraints

$$\max \; v^T x$$

$$Ax \leq c$$

$$\sum_{i \in G_1} x_i = 1$$

$$\vdots$$

$$\sum_{i \in G_k} x_i = 1$$

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

IP (cont'd)

Packing problems.

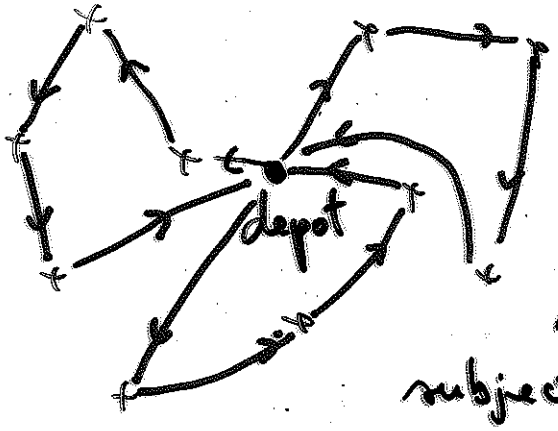
- problems such as knapsack are a particular type of pb so called "packing".

$$\begin{cases} \max\ c^T x \\ Ax \leq b \\ x \in \{0,1\} \end{cases}, \text{ typically } A, b, c^T \geq 0.$$

Covering problems:

$$\begin{cases} \min\ c^T x \\ Ax \geq b \\ x \in \{0,1\} \end{cases}, \text{ typically } A, b, c^T \geq 0$$

Ex: vehicle routing



Given

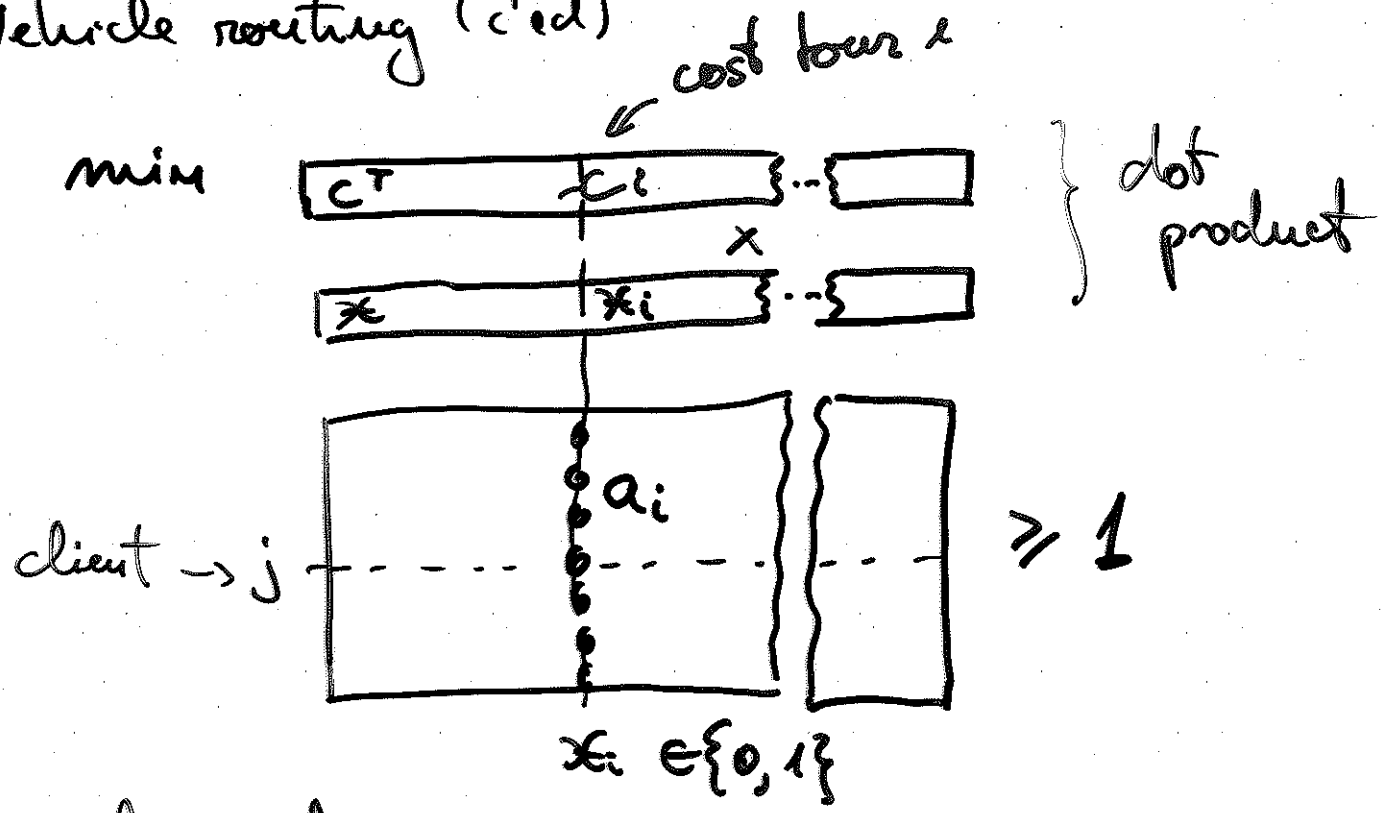
- depot
- x clients

Task: deliver material to clients, subject to truck capacity.

Goal: minimize total distance travelled.

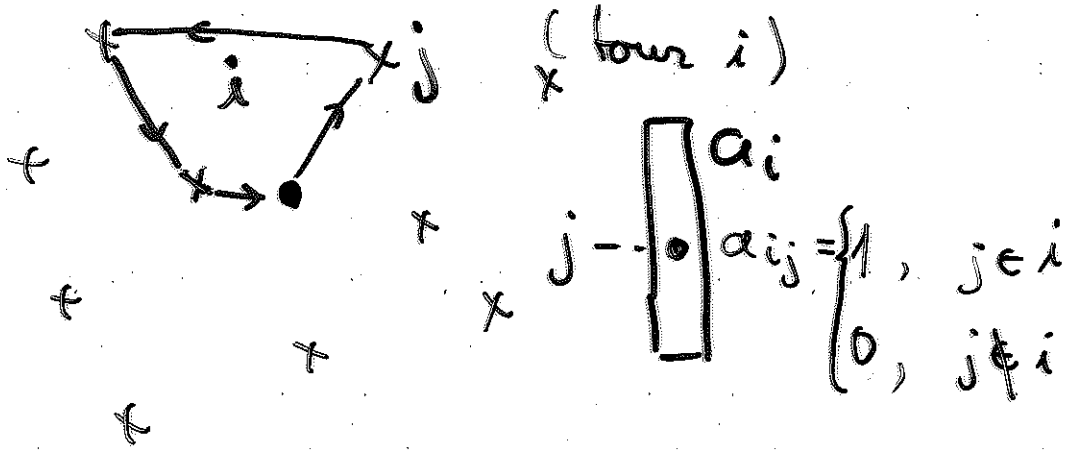
IP (cont'd)

Vehicle routing (cont'd)



Approach

Generate a large number of tours (each obeying truck capacity)

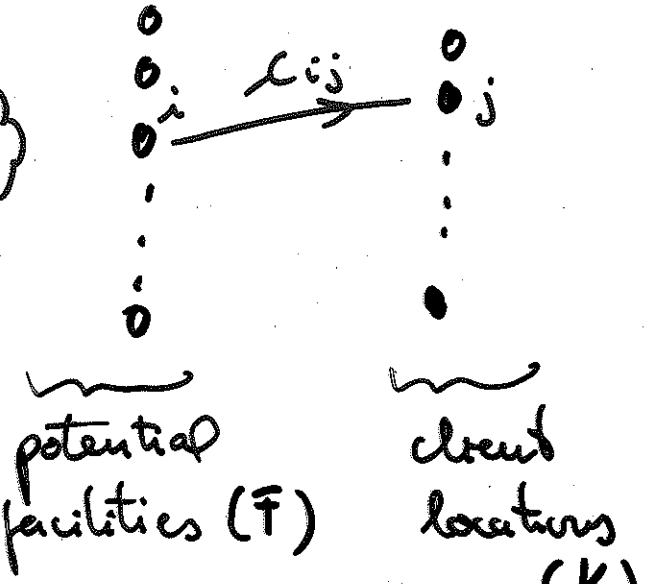


Obs The algorithm is not optimal. However, a variant called "COLUMN GENERATION" is optimal.

IP (cont'd)

Mixed Integer Programs.

Ex: "capacitated facility location problem"



$i \in F$ { f_i : cost of building
a_i : availability (capacity)

$j \in K$ { d_j : demand for products

c_{ij} : cost of transporting a unit of product.

Decision variables:

x_{ij} : amount shipped from i to j ($x_{ij} \in \mathbb{R}$)

$$y_i = \begin{cases} 1, & \text{facility } i \text{ open} \\ 0, & \text{facility } i \text{ not open.} \end{cases}$$

IP (c'ed)

Capacitated facility location (c'ed)

$$\min \sum_{i \in F} f_i y_i + \sum_{\substack{i \in F \\ j \in K}} c_{ij} \cdot x_{ij}$$

$$\sum_{j \in K} x_{ij} \leq a_i \cdot y_i \quad , \forall i \in F$$

(capacity constraints)

$$\sum_{i \in F} x_{ij} \geq d_j \quad , \forall j \in K$$

(demand constraints)

$$y_i \in \{0, 1\}$$

$$x_{ij} \geq 0.$$