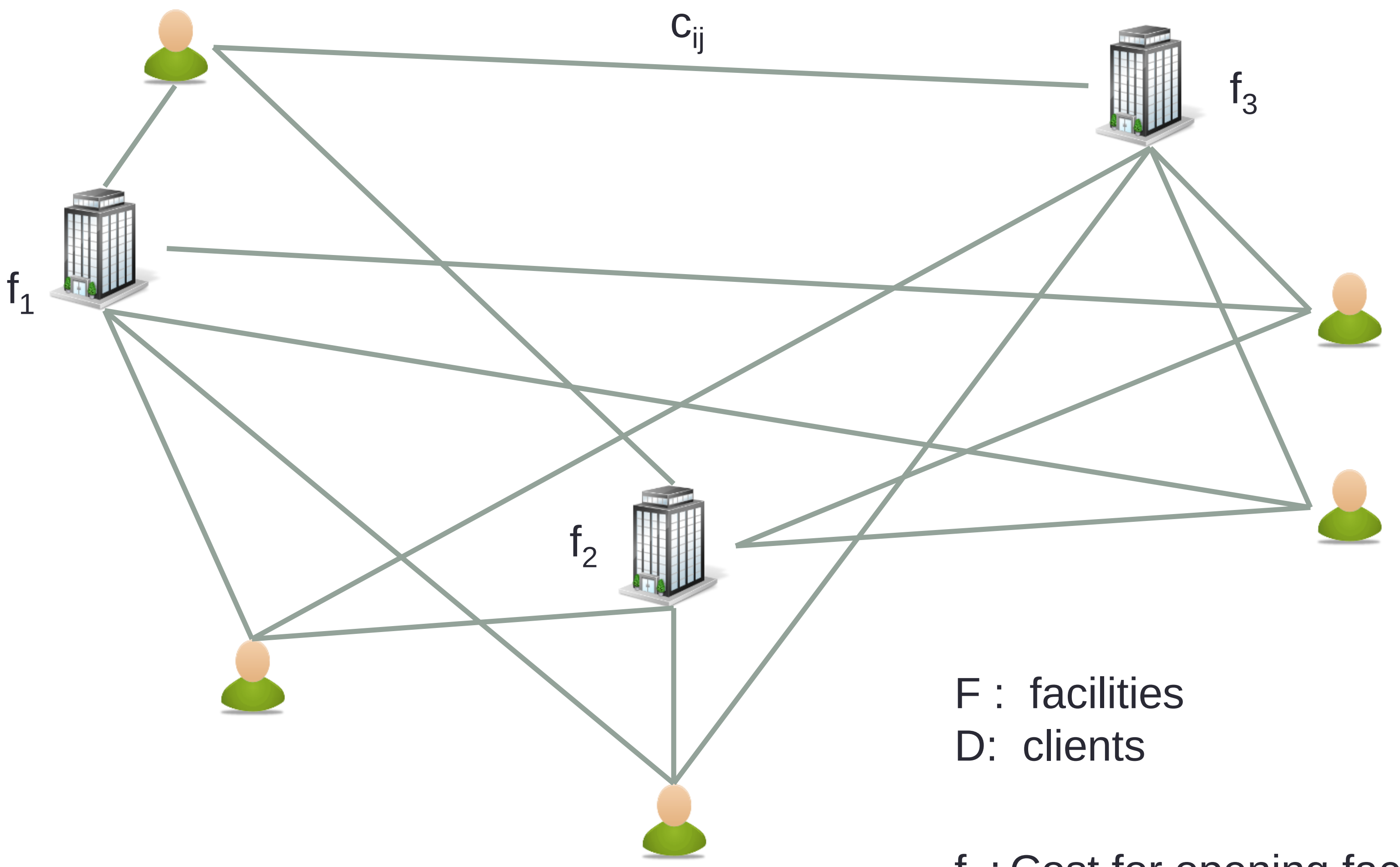
CHAPTER 4

Deterministic rounding of linear programs

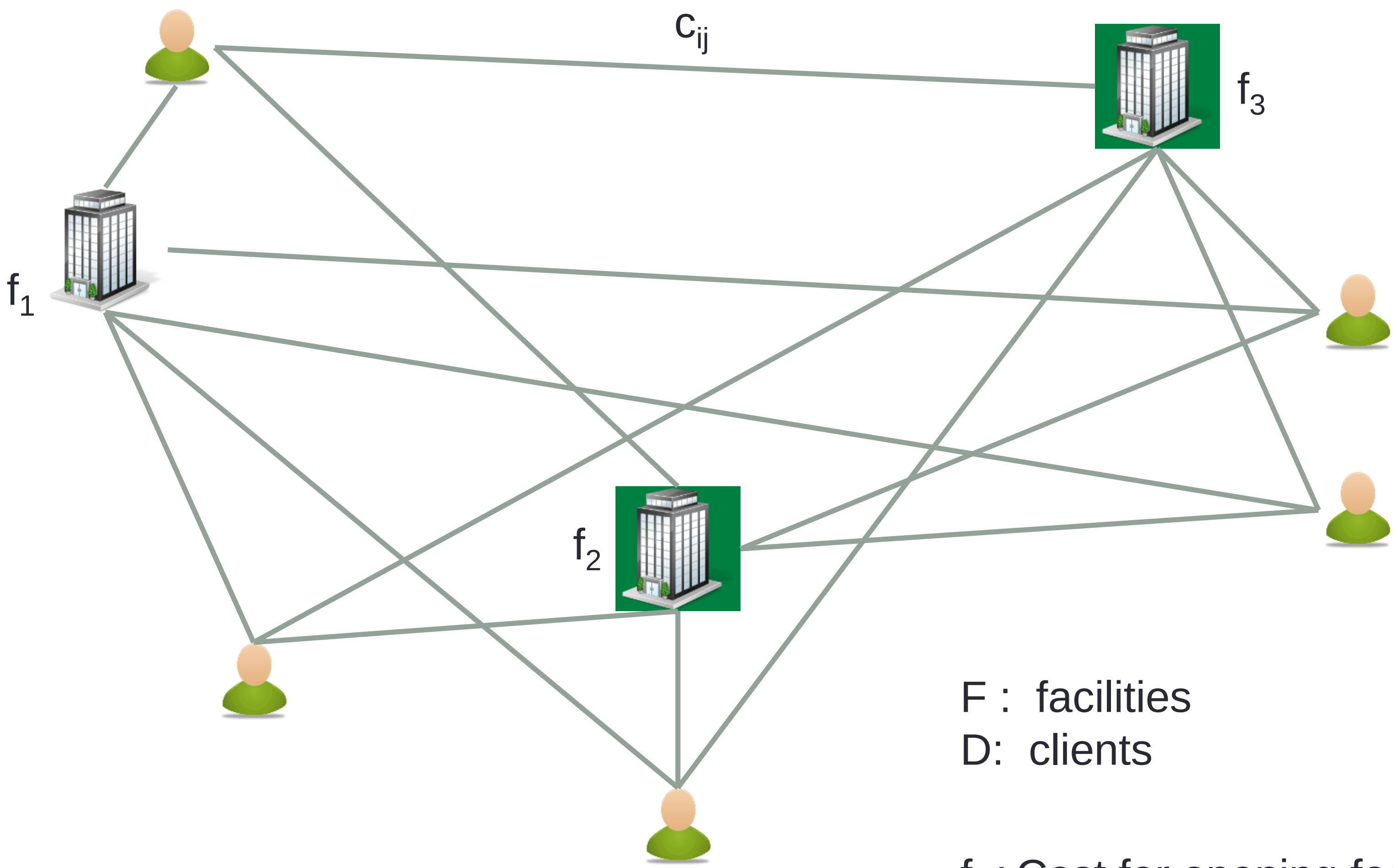
Uncapacitated Facility Location (UFL)



F : facilities
D: clients

f_i : Cost for opening facility i
c_{ij}: Cost for connecting j to i

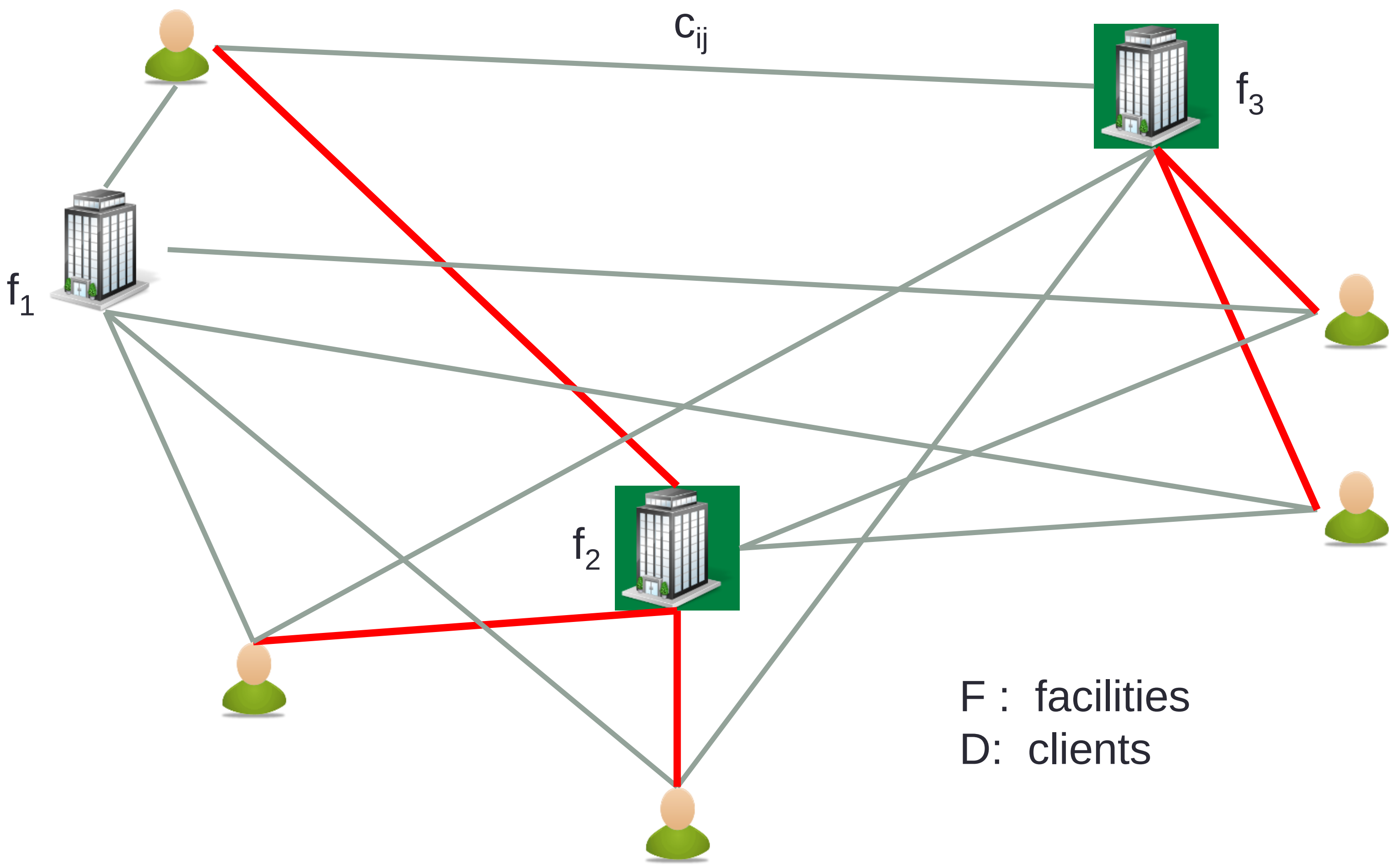
Uncapacitated Facility Location (UFL)



F : facilities
D: clients

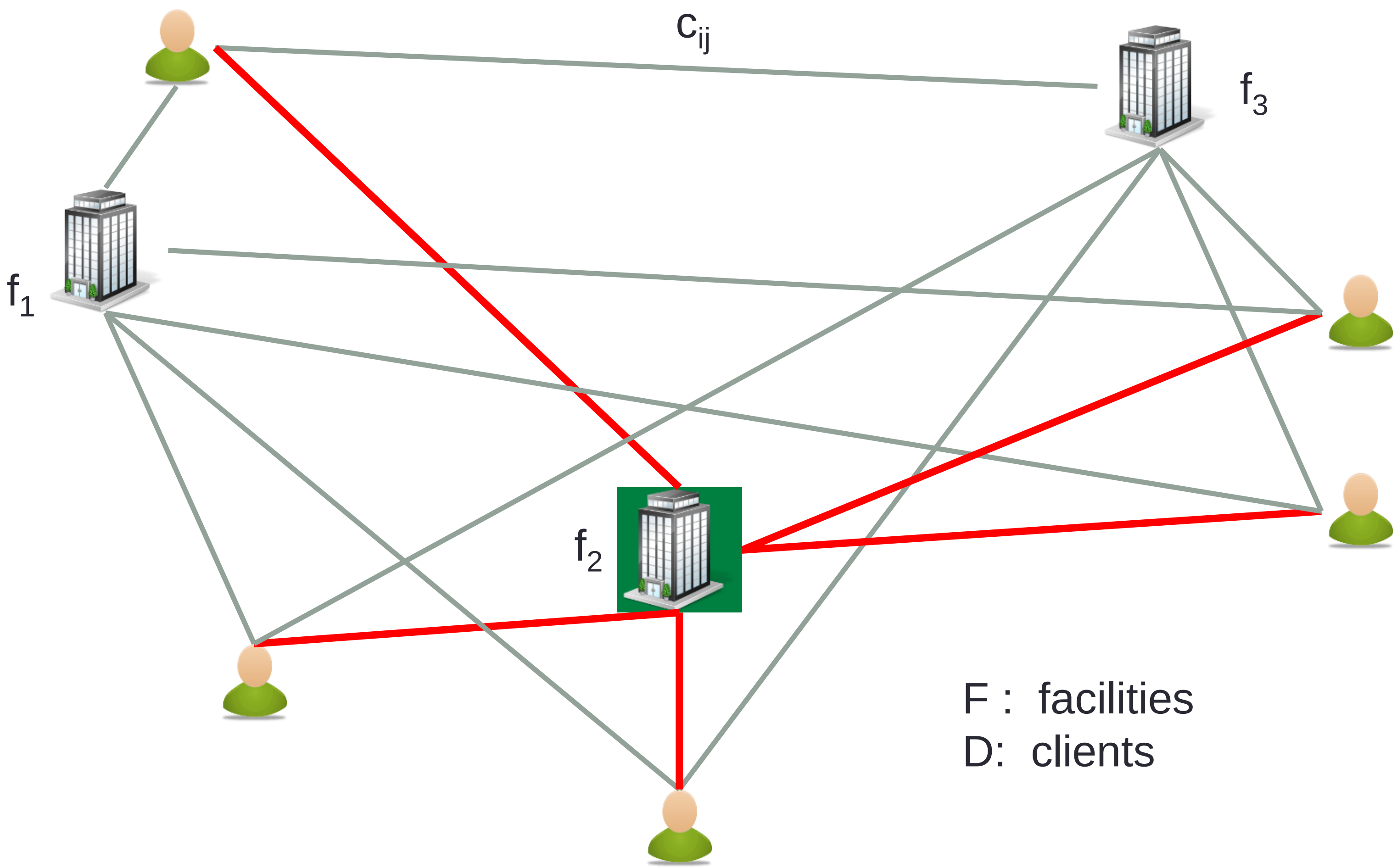
f_i : Cost for opening facility i
c_{ij}: Cost for connecting j to i

Uncapacitated Facility Location (UFL)



Opening cost + Connection cost

Uncapacitated Facility Location (UFL)



F : facilities
D: clients

Opening cost + Connection cost

Uncapacitated Facility Location

$$
\begin{aligned}
\text{(ILP)} \quad \min \quad & Z = \sum_{i \in F} f_i y_i + \sum_{i \in F, j \in D} c_{ij} x_{ij} & \\
s.t. \quad & \sum_{i \in F} x_{ij} = 1 & \text{for all } j \in D, \\
& x_{ij} \leqslant y_i & \text{for all } i \in F, j \in D, \\
& x_{ij} \in \{0, 1\} & \text{for all } i \in F, j \in D, \\
& y_i \in \{0, 1\} & \text{for all } i \in F.
\end{aligned}
$$

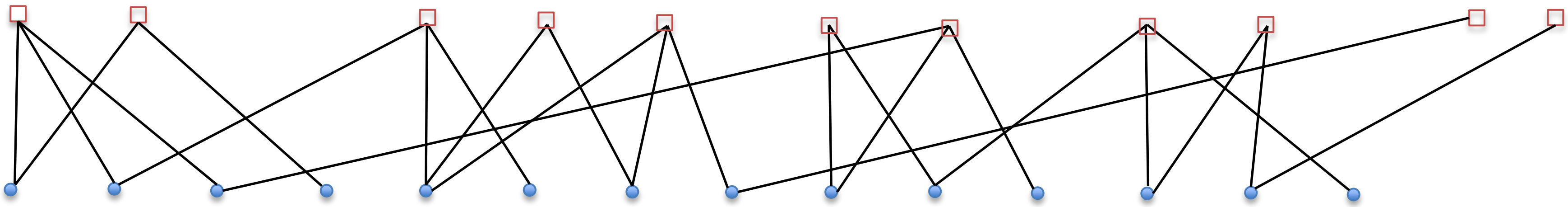
Uncapacitated Facility Location

$$
\begin{array}{llll}
\text{(D)} & \max & Z = \sum_{j \in D} v_j & \\
& s.t. & \sum_{j \in D} w_{ij} \leqslant f_i & \text{for all } i \in F, \\
& & v_j - w_{ij} \leqslant c_{ij} & \text{for all } i \in F, j \in D, \\
& & w_{ij} \geqslant 0 & \text{for all } i \in F, j \in D, \\
& & (v_i \text{ is free}). &
\end{array}
$$

Uncapacitated Facility Location

Solve primal and dual → (x^*,y^*) and (v^*,w^*).

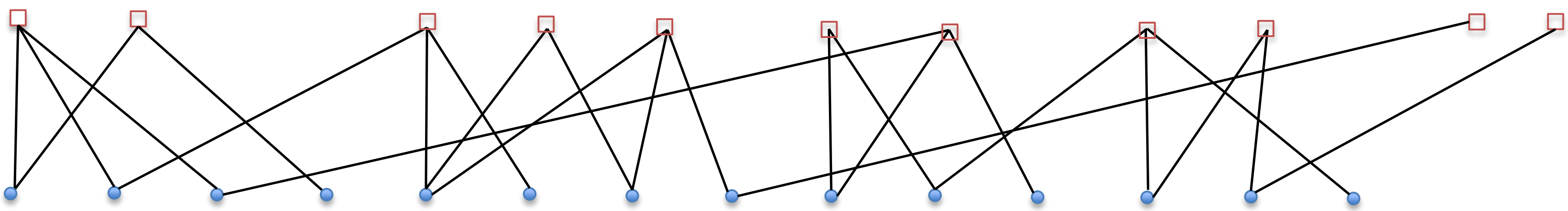
Make *support graph* for x^* : Edge (i,j) if $x_{ij}^*>0$



Uncapacitated Facility Location

Solve primal and dual → (x^*,y^*) and (v^*,w^*).

Make *support graph* for x^* : Edge (i,j) if $x_{ij}^*>0$



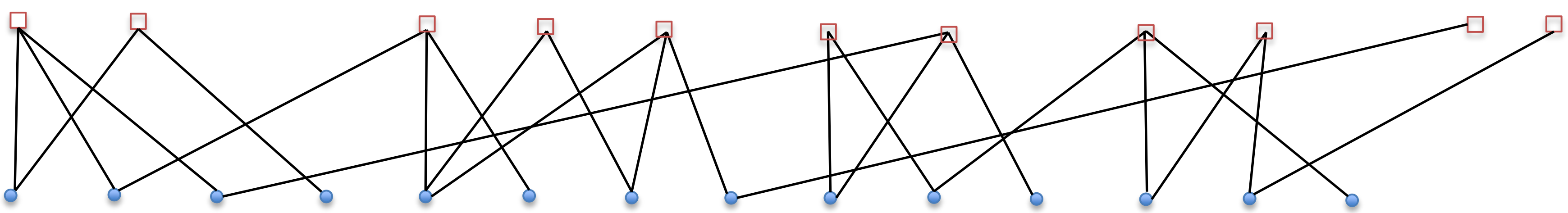
Lemma: If (i,j) in support graph then $v^*_j = c_{ij}+w^*_{ij} \geq c_{ij}$

Proof Follows from complementary slackness and $w_{ij}^* \geq 0$. □

→ Connect each client to an adjacent facility → connection cost $\leq \Sigma_j v_j^*$ ✔

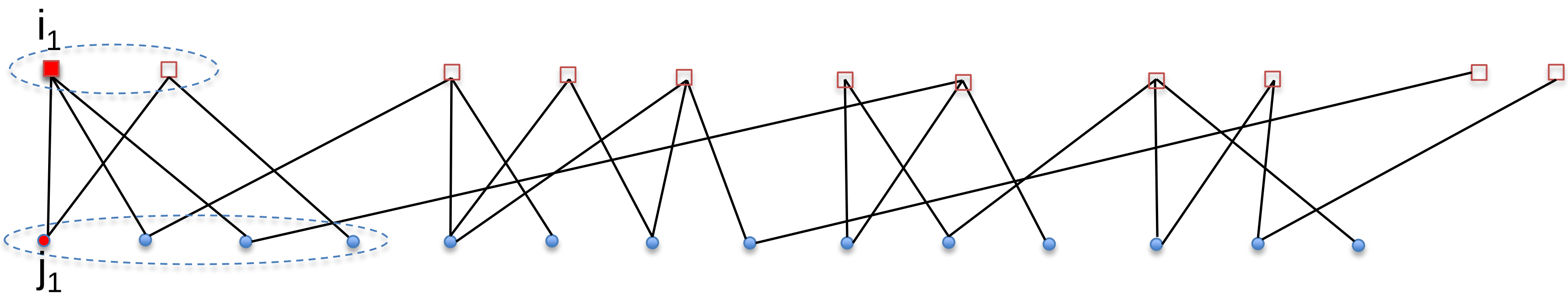
Uncapacitated Facility Location

Algorithm For $k = 1, 2, \ldots$ until all clients are connected do:
Step 1: Among the unconnected clients, choose client j_k with smallest value $v^*_{j_k}$.
Step 2: Choose facility $i_k \in N(j_k)$ with smallest value f_{i_k}.
Step 3: Connect all clients in $N^2(j_k)$ to facility i_k.



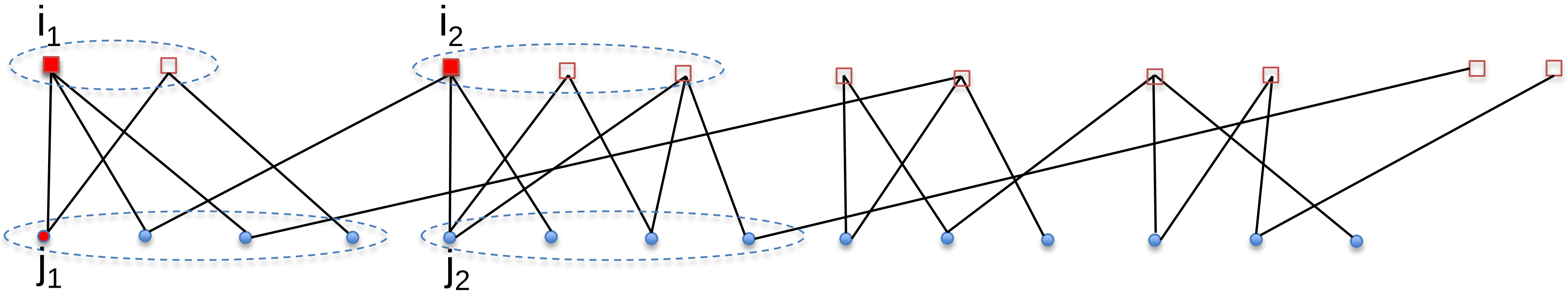
Uncapacitated Facility Location

Algorithm For $k = 1, 2, \ldots$ until all clients are connected do:
Step 1: Among the unconnected clients, choose client j_k with smallest value $v^*_{j_k}$.
Step 2: Choose facility $i_k \in N(j_k)$ with smallest value f_{i_k}.
Step 3: Connect all clients in $N^2(j_k)$ to facility i_k.



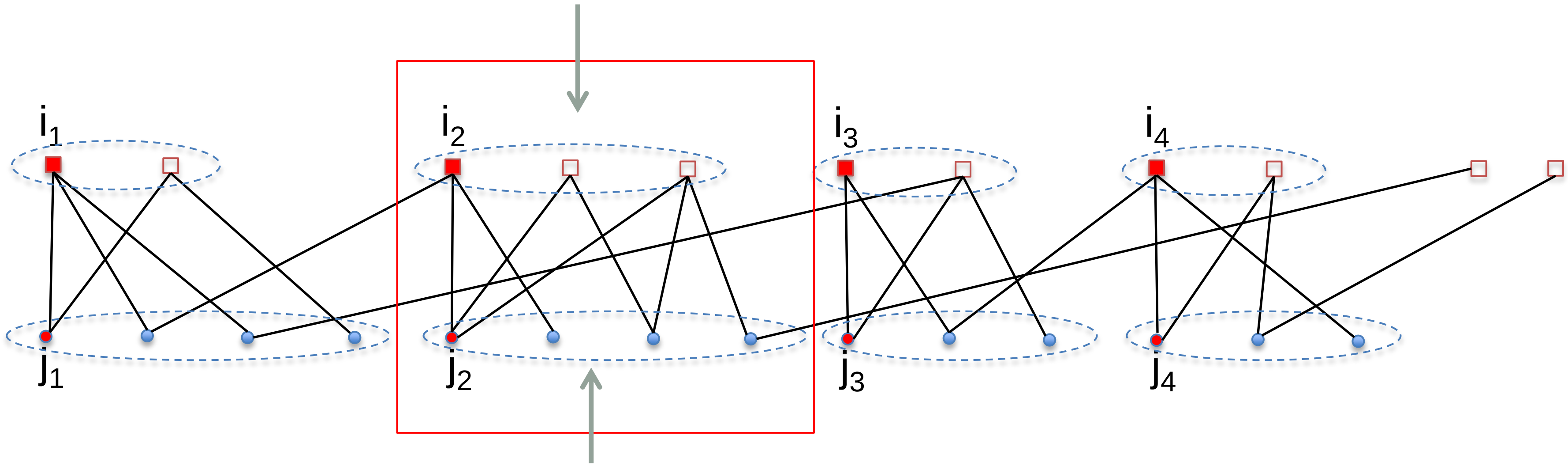
Uncapacitated Facility Location

Algorithm For $k = 1, 2, \ldots$ until all clients are connected do:
Step 1: Among the unconnected clients, choose client j_k with smallest value $v^*_{j_k}$.
Step 2: Choose facility $i_k \in N(j_k)$ with smallest value f_{i_k}.
Step 3: Connect all clients in $N^2(j_k)$ to facility i_k.



Uncapacitated Facility Location

$f(i_2) \leq$ fractional opening cost



connection cost $\leq 3\ \Sigma_j v_j^*$

Total cost $\leq \Sigma_{i \in F} f_i y_i^* + 3\Sigma_{j \in D} v_j^* \leq Z_{LP}^* + 3Z_D^* \leq 4OPT.$

CHAPTER 5

Randomized sampling and
randomized rounding of linear programs

Introduction

Definition
An algorithm ALG for a minimization problem is a randomized α-approximation algorithm if it

1. runs in polynomial time,
2. always finds a feasible solution, and
3. the expected value of the solution is at most α times the optimal value.

- Randomized algorithms are often easier to analyse than deterministic algorithms
- Sometimes, derandomization is possible.

5.1 Max SAT and Max Cut

Max SAT example

$x_1 \vee x_2$, $\neg x_1$, $x_2 \vee \neg x_2 \vee x_3$, $\neg x_2 \vee x_4$, $\neg x_2 \vee \neg x_4$

5 clauses
4 boolean variables x_1, x_2, x_3, x_4
x_1 and $\neg x_1$ are the two literals of variable x_1.

$x_i \in \{\text{TRUE, FALSE}\}$

5.1 Max SAT and Max Cut

Max SAT example

$x_1 \vee x_2$, $\neg x_1$, $x_2 \vee \neg x_2 \vee x_3$, $\neg x_2 \vee x_4$, $\neg x_2 \vee \neg x_4$

5 clauses
4 boolean variables x_1, x_2, x_3, x_4
x_1 and $\neg x_1$ are the two literals of variable x_1.

$x_i \in \{\text{TRUE, FALSE}\}$

Satisfiability problem (SAT):
Is there a true/false assignment such that all clauses are satisfied?

Maximum satisfiability problem (Max SAT):
What is the maximum number of clauses that can be satisfied?

5.1 Max SAT and Max Cut

Max Cut

5.2 Derandomization

Example: Max Cut

- $E[Z]$: expected weight of the cut
- S_i : assignment of $v_1,\ldots,v_i$

Then, $E[Z] = \frac{1}{2} E[\, Z|v_1 \rightarrow U \,] + \frac{1}{2} E[\, Z|v_1 \rightarrow W \,]$

In general, if $v_1,\ldots,v_i$ are already assigned, then

$$E[Z|S_i] = \tfrac{1}{2} E[\, Z|S_i \text{ and } v_{i+1} \rightarrow U \,] + \tfrac{1}{2} E[\, Z|S_i \text{ and } v_{i+1} \rightarrow W \,]$$

Algorithm

For $i=1\ldots n$

Assign v_i to the side (U or W) with largest expected value.

5.2 Derandomization

Theorem
Derandomized algorithm is a ½-approximation for Max Cut

Proof

$E[Z|S_{i-1}] = ½\ E[\ Z|S_{i-1}$ and $v_i \rightarrow U\ ] + ½\ E[\ Z|S_{i-1}$ and $v_i \rightarrow W\ ]$

$\rightarrow\ E[Z|S_i] \geq E[Z|S_{i-1}]$ for all i.

Value of solution is $E[Z|S_n]$

$E[Z|S_n] \geq E[Z|S_{n-1}] \geq \ldots \geq E[Z|S_1] \geq E[Z] \geq OPT/2$

5.2 Derandomization

Derandomized algorithm

For i=1…n

Assign v_i to the side that adds the largest weight to the cut.

5.2 Derandomization

`**Method of conditional expectations**’

- Not always possible.

May be possible if
- algorithm makes a number of independent random decisions

Sometimes,
- computing conditional expections is difficult / not possible.

5.2 Derandomization

Example: Max SAT

- E[Z]: expected number of clauses satisfied
- S_i : assignment of $x_1,\dots,x_i$

Then, $E[Z] = \frac{1}{2} E[\, Z|x_1=\text{true}\,] + \frac{1}{2} E[\, Z|x_1=\text{false}\,]$

In general, if $x_1,\dots,x_i$ are already assigned, then

$E[Z|S_{i-1}] = \frac{1}{2} E[\, Z|S_{i-1} \text{ and } x_i=\text{true}] + \frac{1}{2} E[\, Z|S_{i-1} \text{ and } x_i=\text{false}\,]$

Algorithm
For i=1…n
Set x_i = true if $E[\, Z|S_{i-1} \text{ and } x_i=\text{true}] \geq E[\, Z|S_{i-1} \text{ and } x_i=\text{false}\,]$ and set x_i=false otherwise.

5.3 Biased coin flipping

(unweighted)

Algorithm

1. If the set of clauses contains a clause $\neg x_i$ but does not contain a clause x_i then, in every clause replace $\neg x_i$ by x_i and vice versa.
2. Set each variable independently at random to true with probability p.

Theorem

For $p=p^*=(\sqrt{5}-1)/2$, algorithm is a randomzied $p^*\approx 0.62$-approximation.