The Maximum Facility Location Problem

Mitchell Jones

Supervisor: Julián Mestre

September 8, 2015

THE UNIVERSITY OF SYDNEY

OVERVIEW

MAXIMUM FACILITY LOCATION

- Assign clients to facilities at certain cost
- Each facility has an interval
- Open facilities can not overlap
- Goal: Select a set of non-overlapping facilities to maximise cost of assignments

Maximum Facility Location

- Assign clients to facilities at certain cost
- Each facility has an interval
- Open facilities can not overlap
- Goal: Select a set of non-overlapping facilities to maximise cost of assignments

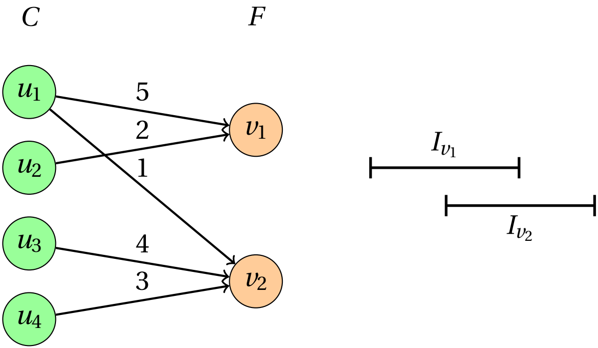
MAXIMUM FACILITY LOCATION

- Assign clients to facilities at certain cost
- Each facility has an interval
- Open facilities can not overlap
- Goal: Select a set of non-overlapping facilities to maximise cost of assignments

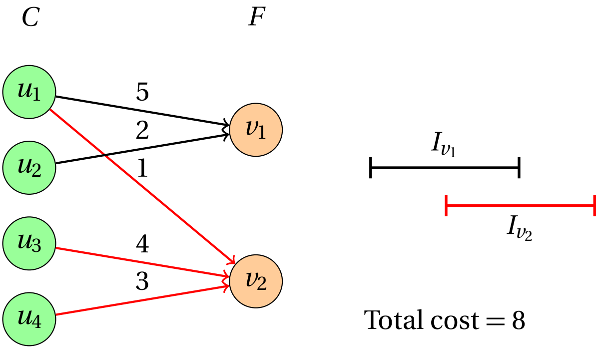
Maximum Facility Location

- Assign clients to facilities at certain cost
- Each facility has an interval
- Open facilities can not overlap
- Goal: Select a set of non-overlapping facilities to maximise cost of assignments

Example



EXAMPLE (CONT.)



Total cost $= 8$

Overview

Problem

Notation & Definitions

- Notation
- Definitions
- Submodular Functions

Integer Program

Independent Rounding

Dependent Rounding

Conclusion

Notation

- C is the set of clients
- F is the set of facilities
- The function $w : C \times F \to \mathbb{R}$ is the weight of assigning a client to a facility
- Each facility $v \in F$ has an interval I_v on the real line
- P is the set of endpoints of all intervals

NOTATION

- C is the set of clients
- F is the set of facilities
- The function $w: C \times F \to \mathbb{R}$ is the weight of assigning a client to a facility
- Each facility $v \in F$ has an interval I_v on the real line
- P is the set of endpoints of all intervals

Notation

- C is the set of clients
- F is the set of facilities
- The function $w: C \times F \to \mathbb{R}$ is the weight of assigning a client to a facility
- Each facility $v \in F$ has an interval I_v on the real line
- P is the set of endpoints of all intervals

Notation

- C is the set of clients
- F is the set of facilities
- The function $w: C \times F \to \mathbb{R}$ is the weight of assigning a client to a facility
- Each facility $v \in F$ has an interval I_v on the real line
- P is the set of endpoints of all intervals

Definitions

- A subset of facilities $S \subseteq F$ is *independent* if no two intervals in $\{I_v : v \in S\}$ overlap.
- Define a set function which produces a real value for any $S \subseteq F$

$$f(S) = \sum_{u \in C} \max_{v \in S} w_{uv} \tag{1}$$

- An instance of the *maximum facility location problem* is given by (C, F, w, I)
- Goal: Select an independent set $S \subseteq F$ maximising $f(S)$

Definitions

- A subset of facilities $S \subseteq F$ is *independent* if no two intervals in $\{I_v : v \in S\}$ overlap.
- Define a set function which produces a real value for any $S \subseteq F$

$$f(S) = \sum_{u \in C} \max_{v \in S} w_{uv} \tag{1}$$

- An instance of the *maximum facility location problem* is given by (C, F, w, I)
- Goal: Select an independent set $S \subseteq F$ maximising $f(S)$

DEFINITIONS

- A subset of facilities $S \subseteq F$ is *independent* if no two intervals in $\{I_v : v \in S\}$ overlap.
- Define a set function which produces a real value for any $S \subseteq F$

$$f(S) = \sum_{u \in C} \max_{v \in S} w_{uv} \tag{1}$$

- An instance of the *maximum facility location problem* is given by (C, F, w, I)
- Goal: Select an independent set $S \subseteq F$ maximising $f(S)$

Definitions

- A subset of facilities $S \subseteq F$ is *independent* if no two intervals in $\{I_v : v \in S\}$ overlap.
- Define a set function which produces a real value for any $S \subseteq F$

$$f(S) = \sum_{u \in C} \max_{v \in S} w_{uv} \tag{1}$$

- An instance of the *maximum facility location problem* is given by (C, F, w, I)
- Goal: Select an independent set $S \subseteq F$ maximising $f(S)$

SUBMODULAR FUNCTIONS

Definition

A set function f is submodular if $\forall A \subseteq B \subseteq F$ and $\forall v \in F \setminus B$

$$f(A+v) - f(A) \geq f(B+v) - f(B).$$

Definition

A set function f is non-decreasing if for any $A \subseteq B \subseteq F$

$$f(B) \geq f(A).$$

Lemma

The function f in Equation 1 is non-decreasing and submodular.

Overview

Modelling Constraints

- Create two variables:
  1. A client-facility variable $x_{uv} = 1$ if we assign a client $u \in C$ to a facility $v \in F$ and 0 otherwise
  2. A facility variable $y_v = 1$ for a facility $v \in F$ if we open v and 0 otherwise
- Three constraints that we need to model:
  1. A client can be assigned to at most one facility
  2. If we assign a client to a facility, the facility must be open
  3. Open facilities cannot overlap

MODELLING CONSTRAINTS

- Create two variables:
  1. A client-facility variable $x_{uv} = 1$ if we assign a client $u \in C$ to a facility $v \in F$ and 0 otherwise
  2. A facility variable $y_v = 1$ for a facility $v \in F$ if we open v and 0 otherwise
- Three constraints that we need to model:
  1. A client can be assigned to at most one facility
  2. If we assign a client to a facility, the facility must be open
  3. Open facilities cannot overlap

MODELLING CONSTRAINTS

- Create two variables:
  1. A client-facility variable $x_{uv} = 1$ if we assign a client $u \in C$ to a facility $v \in F$ and 0 otherwise
  2. A facility variable $y_v = 1$ for a facility $v \in F$ if we open v and 0 otherwise
- Three constraints that we need to model:
  1. A client can be assigned to at most one facility
  2. If we assign a client to a facility, the facility must be open
  3. Open facilities cannot overlap

MODELLING CONSTRAINTS

- Create two variables:
  1. A client-facility variable $x_{uv} = 1$ if we assign a client $u \in C$ to a facility $v \in F$ and 0 otherwise
  2. A facility variable $y_v = 1$ for a facility $v \in F$ if we open v and 0 otherwise
- Three constraints that we need to model:
  1. A client can be assigned to at most one facility
  2. If we assign a client to a facility, the facility must be open
  3. Open facilities cannot overlap

MODELLING CONSTRAINTS

- Create two variables:
  1. A client-facility variable $x_{uv} = 1$ if we assign a client $u \in C$ to a facility $v \in F$ and 0 otherwise
  2. A facility variable $y_v = 1$ for a facility $v \in F$ if we open v and 0 otherwise
- Three constraints that we need to model:
  1. A client can be assigned to at most one facility
  2. If we assign a client to a facility, the facility must be open
  3. Open facilities cannot overlap

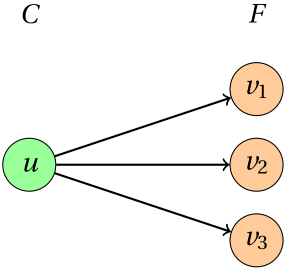
MODELLING CONSTRAINTS

- Create two variables:
  1. A client-facility variable $x_{uv} = 1$ if we assign a client $u \in C$ to a facility $v \in F$ and 0 otherwise
  2. A facility variable $y_v = 1$ for a facility $v \in F$ if we open v and 0 otherwise
- Three constraints that we need to model:
  1. A client can be assigned to at most one facility
  2. If we assign a client to a facility, the facility must be open
  3. Open facilities cannot overlap

Modelling Constraints

- Create two variables:
  1. A client-facility variable $x_{uv} = 1$ if we assign a client $u \in C$ to a facility $v \in F$ and 0 otherwise
  2. A facility variable $y_v = 1$ for a facility $v \in F$ if we open v and 0 otherwise
- Three constraints that we need to model:
  1. A client can be assigned to at most one facility
  2. If we assign a client to a facility, the facility must be open
  3. Open facilities cannot overlap

1. A client can be assigned to at most one facility



$x_{uv_1} + x_{uv_2} + x_{uv_3} \leq 1$

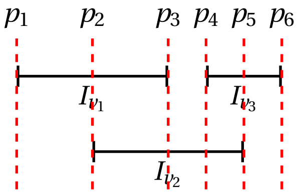
More generally:

$\sum_{v \in F} x_{uv} \leq 1$

2. If we assign a client to a facility, the facility must be open

- If we want to assign u to v, then $x_{uv} = 1$
- But v must be open, therefore $y_v = 1$
- Therefore: $x_{uv} \leq y_v$

3. Open facilities cannot overlap



- We require $y_{v_1} + y_{v_2} \leq 1$ and $y_{v_2} + y_{v_3} \leq 1$
- In general, for each endpoint $p \in P$

$$\sum_{v \in F : p \in I_v} y_v \leq 1$$

Integer Program

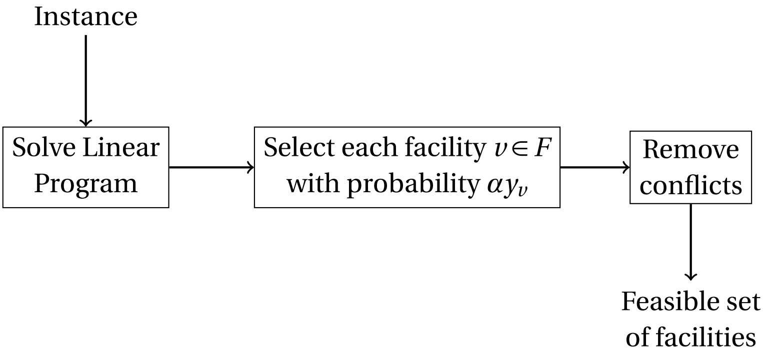
$$
\begin{array}{llllll}
\text{maximise} & \displaystyle\sum_{u\in C, v\in F} w_{uv}x_{uv} & & & & \\
\text{subject to} & \displaystyle\sum_{v\in F} x_{uv} & \leq & 1 & \forall u\in C & \\
 & x_{uv} & \leq & y_v & \forall u\in C, v\in F & \text{(IP)} \\
 & \displaystyle\sum_{v\in F: p\in I_v} y_v & \leq & 1 & \forall p\in P & \\
 & x_{uv} & \in & \{0,1\} & \forall u\in C, v\in F & \\
 & y_v & \in & \{0,1\} & \forall v\in F &
\end{array}
$$

Linear Program

$$
\begin{array}{llll}
\text{maximise} & \displaystyle\sum_{u\in C, v\in F} w_{uv}x_{uv} & & \\
\text{subject to} & \displaystyle\sum_{v\in F} x_{uv} \leq 1 & \forall u\in C & \\
& x_{uv} \leq y_v & \forall u\in C, v\in F & \\
& \displaystyle\sum_{v\in F: p\in I_v} y_v \leq 1 & \forall p\in P & \\
& x_{uv} \geq 0 & \forall u\in C, v\in F & \\
& y_v \geq 0 & \forall v\in F &
\end{array}
\tag{LP}
$$

Overview

Algorithm


Instance
Solve Linear Program
Select each facility $v \in F$ with probability αy_v
Remove conflicts
Feasible set of facilities

ALGORITHM

Algorithm 1 Independent rounding with alterations.

function SELECT-AND-FILTER(α)

$(x, y) \leftarrow$ solve (LP)

$R \leftarrow$ Select each $v \in F$ with probability αy_v

$T \leftarrow \{ v \in R : \nexists v' \in R$ such that start point of I_v lies in $I_{v'} \}$

return T

end function

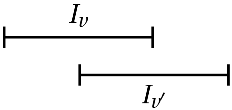
RESULTS

Lemma

The set T returned by SELECT-AND-FILTER *is independent.*

Proof.

- Suppose T is not independent.
- Assume w.l.o.g. that I_v lies to the left of $I_{v'}$.
- Since $v' \in T$, then $v' \in R$.
- Then v' cannot be in T, since I_v intersects the start point of $I_{v'}$.



□

RESULTS (CONT.)

Lemma

Let R be the set sampled by SELECT-AND-FILTER, *then*

$$\mathbb{E}[f(R)] \geq (1 - e^{-\alpha}) \sum_{u \in C, v \in F} w_{uv} x_{uv}.$$

Lemma

Let R and T be the sets computed in SELECT-AND-FILTER *then*

$$\mathbb{E}[f(T)] \geq (1 - \alpha) \mathbb{E}[f(R)].$$

Final Result

Theorem

There is an LP-rounding 0.19-approximation for maximum facility location.

Proof.

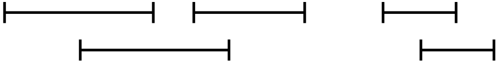
We know T is independent. The expected value of T is

$$\mathbb{E}[f(T)] \geq (1-\alpha)\mathbb{E}[f(R)] \geq (1-\alpha)(1-e^{-\alpha}) \sum_{u\in C, v\in F} w_{uv}x_{uv}.$$

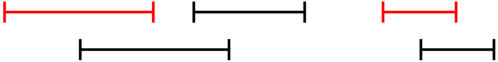
The best approximation ratio is attained at $\alpha = 0.44$, where we get 0.199. $\square$

Removing Conflicts

Consider the following scenario:



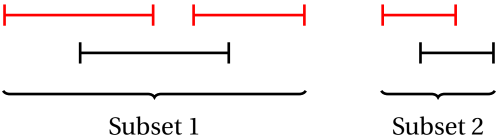
The current conflict resolution scheme selects two intervals:



But we can actually pick three!

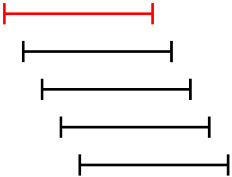
Removing Conflicts

1. Split the intervals into disjoint subsets (compute the union of the intervals)
2. For each subset, select a maximal subset (greedily) starting with the left most



Removing Conflicts

However we don't get a better guarantee...



Overview

Algorithm

- Gradually shift the fractional (LP) solution to an integral one
- Let $S = \{ v \in F : 0 < y_v < 1 \}$ be the set of facilities with a non-integral value
- For any $p \in P$, the slack(p) is the slack of the corresponding constraint in (LP)

$$\text{slack}(p) = 1 - \sum_{v \in F : p \in I_v} y_v$$

- The point p is *tight* if slack$(p) = 0$

Algorithm

- Gradually shift the fractional (LP) solution to an integral one
- Let $S = \{ v \in F : 0 < y_v < 1 \}$ be the set of facilities with a non-integral value
- For any $p \in P$, the slack(p) is the slack of the corresponding constraint in (LP)

$$\text{slack}(p) = 1 - \sum_{v \in F : p \in I_v} y_v$$

- The point p is *tight* if $\text{slack}(p) = 0$

Algorithm

- Gradually shift the fractional (LP) solution to an integral one
- Let $S = \{ v \in F : 0 < y_v < 1 \}$ be the set of facilities with a non-integral value
- For any $p \in P$, the slack(p) is the slack of the corresponding constraint in (LP)

$$\text{slack}(p) = 1 - \sum_{v \in F : p \in I_v} y_v$$

- The point p is *tight* if $\text{slack}(p) = 0$

ALGORITHM

- Gradually shift the fractional (LP) solution to an integral one
- Let $S = \{ v \in F : 0 < y_v < 1 \}$ be the set of facilities with a non-integral value
- For any $p \in P$, the slack(p) is the slack of the corresponding constraint in (LP)

$$\text{slack}(p) = 1 - \sum_{v \in F : p \in I_v} y_v$$

- The point p is *tight* if slack(p) $= 0$

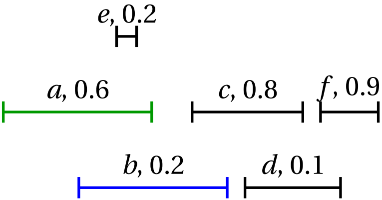
Algorithm

1. Select two disjoint independent subsets $M_1, M_2 \subset S$
2. Update the y-values of $M_1 \cup M_2$ randomly so that either:
   - A facility variable is integral; or
   - A point is tight
3. Repeat the procedure until all facility variables are integral

SELECTING M_1 AND M_2

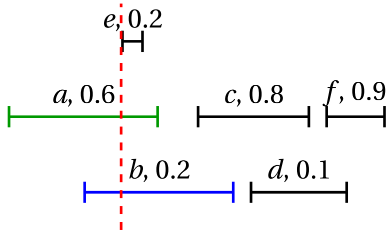
1. Let $\widetilde{P}$ be the set of tight points from left to right
2. Let v_1 and v_2 be the first two facilities
3. Set $M_1 = \{v_1\}$ and $M_2 = \{v_2\}$
4. For every tight point $t \in \widetilde{P}$
   - 4.1 If I_{v_1} contains t and I_{v_2} contains t then do nothing
   - 4.2 Otherwise, let v be the facility starting at t
   - 4.3 Add v to M_1 (or M_2) if t stabs I_{v_2} (or I_{v_1})
   - 4.4 Update $v_1 = v$ (or $v_2 = v$)

Example



$M_1 = \{a\}$
$M_2 = \{b\}$

Example

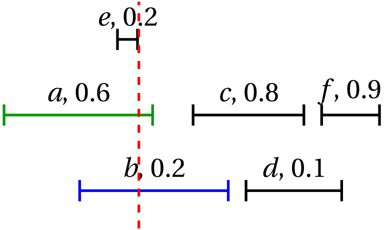


$M_1 = \{a\}, v_1 = a$

$M_2 = \{b\}, v_2 = b$

Do nothing

Example

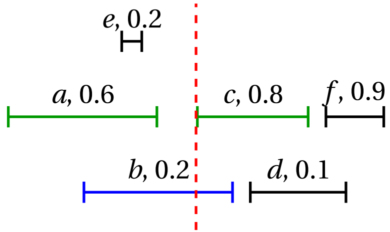


$M_1 = \{a\}, v_1 = a$

$M_2 = \{b\}, v_2 = b$

Do nothing

Example

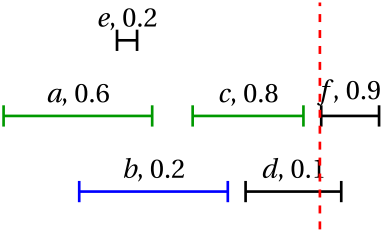


$M_1 = \{a, c\}, v_1 = v = c$

$M_2 = \{b\}, v_2 = b$

Intersects I_{v_2} (I_b), add I_c to M_1 and update $v_1 = c$

Example



$M_1 = \{a, c\}, v_1 = c$
$M_2 = \{b\}, v_2 = b$

Neither I_{v_1} (I_c) or I_{v_2} (I_b) intersect the tight point, terminate

Rounding the Solution

- For $i = 1, 2$, let $P_i \subseteq P$ be the points that intersect an interval in M_i
- Compute the following

$$\varepsilon = \min\left(\min_{p \in P_1 \setminus P_2} \text{slack}(p), \min_{v \in M_2} y_v\right) \qquad (\uparrow M_1 \quad \downarrow M_2)$$

$$\delta = \min\left(\min_{p \in P_2 \setminus P_1} \text{slack}(p), \min_{v \in M_1} y_v\right) \qquad (\downarrow M_1 \quad \uparrow M_2)$$

Rounding the Solution

- For $i = 1, 2$, let $P_i \subseteq P$ be the points that intersect an interval in M_i
- Compute the following

$$\varepsilon = \min\left(\min_{p \in P_1 \setminus P_2} \mathrm{slack}(p), \min_{v \in M_2} y_v\right) \qquad (\uparrow M_1 \quad \downarrow M_2)$$

$$\delta = \min\left(\min_{p \in P_2 \setminus P_1} \mathrm{slack}(p), \min_{v \in M_1} y_v\right) \qquad (\downarrow M_1 \quad \uparrow M_2)$$

Rounding the Solution

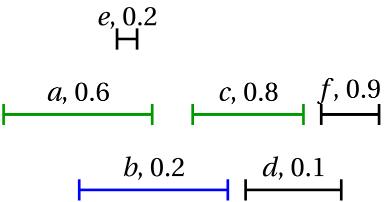
With probability $\frac{\varepsilon}{\varepsilon+\delta}$, we update y to be

$$y'_v = \begin{cases} y_v - \delta, v \in M_1, \\ y_v + \delta, v \in M_2, \\ y_v, \text{ otherwise.} \end{cases}$$

Otherwise with complimentary probability $\frac{\delta}{\varepsilon+\delta}$, replace y with

$$y''_v = \begin{cases} y_v + \varepsilon, v \in M_1, \\ y_v - \varepsilon, v \in M_2, \\ y_v, \text{otherwise.} \end{cases}$$

Example (cont.)



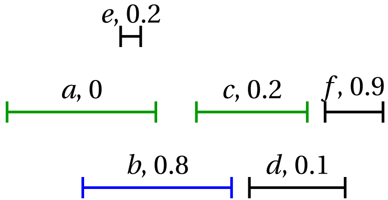
$M_1 = \{a, c\}$

$M_2 = \{b\}$

$\varepsilon = 0.1, \delta = 0.6$

Assume we decrease M_1 by δ and increase M_2 by δ

Example (cont.)



$M_1 = \{a, c\}$
$M_2 = \{b\}$

Finished iteration,
a is now an integral value

Results

Lemma

M_1 and M_2 are disjoint and both are independent. Furthermore, any tight point p either stabs both M_1 and M_2, or none of them.

Lemma

In each iteration we either gain a new integral facility or new tight point.

Lemma

The new solution at the end of each iteration (y' or y'') obeys the independence and non-negativity constraints of (LP).

RESULTS

Lemma

Let y be the fractional solution at the beginning, and $\hat{y}$ be the solution at the end of the iteration. Then for every $v \in F$ we have $\mathbb{E}[\hat{y}_v] = y_v$.

Proof.

- If a facility $v \in F$ is not in M_1 or M_2, then the equality holds.
- For each facility $v \in M_1$ we have

$$\begin{aligned} \mathbb{E}[\hat{y}_v] &= \left(\frac{\varepsilon}{\varepsilon+\delta}\right)(y_v - \delta) + \left(\frac{\delta}{\varepsilon+\delta}\right)(y_v + \varepsilon) \\ &= \frac{\varepsilon y_v - \varepsilon\delta + \delta y_v + \varepsilon\delta}{\varepsilon+\delta} = y_v. \end{aligned}$$

- A similar argument applies for facilities $v \in M_2$.

□

Final Result

Theorem

Let T be the set returned by the dependent rounding algorithm. Then T is independent and $\Pr[v \in T] = y_v$ *for all* $v \in F$*.*

Proof.

- By the previous lemma, at each iteration we either gain an additional integral facility or tight point.
- After a finite amount of iterations, we terminate with an integral solution $\hat{y}$.
- We maintain feasibility at each iteration, therefore $T = \{ v \in F : \hat{y}_v = 1 \}$.
- By induction and the expectation of $\hat{y}$, we have that $\Pr[v \in T] = y_v$ for all $v \in F$.

□

Final Result

Theorem

Let T be the set returned by the dependent rounding algorithm. Then T is independent and $\Pr[v \in T] = y_v$ *for all* $v \in F$*.*

Proof.

- By the previous lemma, at each iteration we either gain an additional integral facility or tight point.
- After a finite amount of iterations, we terminate with an integral solution $\hat{y}$.
- We maintain feasibility at each iteration, therefore $T = \{ v \in F : \hat{y}_v = 1 \}$.
- By induction and the expectation of $\hat{y}$, we have that $\Pr[v \in T] = y_v$ for all $v \in F$.

$\square$

Final Result

Theorem

Let T be the set returned by the dependent rounding algorithm. Then T is independent and $\Pr[v \in T] = y_v$ for all $v \in F$.

Proof.

- By the previous lemma, at each iteration we either gain an additional integral facility or tight point.
- After a finite amount of iterations, we terminate with an integral solution $\hat{y}$.
- We maintain feasibility at each iteration, therefore $T = \{ v \in F : \hat{y}_v = 1 \}$.
- By induction and the expectation of $\hat{y}$, we have that $\Pr[v \in T] = y_v$ for all $v \in F$.

$\square$

Final Result

Theorem

Let T be the set returned by the dependent rounding algorithm. Then T is independent and $\Pr[v \in T] = y_v$ for all $v \in F$.

Proof.

- By the previous lemma, at each iteration we either gain an additional integral facility or tight point.
- After a finite amount of iterations, we terminate with an integral solution $\hat{y}$.
- We maintain feasibility at each iteration, therefore $T = \{ v \in F : \hat{y}_v = 1 \}$.
- By induction and the expectation of $\hat{y}$, we have that $\Pr[v \in T] = y_v$ for all $v \in F$.

$\square$

Overview

Concluding Statements

- Modelled the maximum facility location problem as an integer/linear program
- Presented a 0.19-approximation algorithm
- Discussed a dependent rounding scheme
- Future work:
  - Improving the 0.19-approximation algorithm's guarantee
  - Analysing the guarantee of the dependent rounding scheme
  - New conflict resolution heuristics

Concluding Statements

- Modelled the maximum facility location problem as an integer/linear program
- Presented a 0.19-approximation algorithm
- Discussed a dependent rounding scheme
- Future work:
  - Improving the 0.19-approximation algorithm's guarantee
  - Analysing the guarantee of the dependent rounding scheme
  - New conflict resolution heuristics

Concluding Statements

- Modelled the maximum facility location problem as an integer/linear program
- Presented a 0.19-approximation algorithm
- Discussed a dependent rounding scheme
- Future work:
  - Improving the 0.19-approximation algorithm's guarantee
  - Analysing the guarantee of the dependent rounding scheme
  - New conflict resolution heuristics

Concluding Statements

- Modelled the maximum facility location problem as an integer/linear program
- Presented a 0.19-approximation algorithm
- Discussed a dependent rounding scheme
- Future work:
  - Improving the 0.19-approximation algorithm's guarantee
  - Analysing the guarantee of the dependent rounding scheme
  - New conflict resolution heuristics

Concluding Statements

- Modelled the maximum facility location problem as an integer/linear program
- Presented a 0.19-approximation algorithm
- Discussed a dependent rounding scheme
- Future work:
  - Improving the 0.19-approximation algorithm's guarantee
  - Analysing the guarantee of the dependent rounding scheme
  - New conflict resolution heuristics

Concluding Statements

- Modelled the maximum facility location problem as an integer/linear program
- Presented a 0.19-approximation algorithm
- Discussed a dependent rounding scheme
- Future work:
  - Improving the 0.19-approximation algorithm's guarantee
  - Analysing the guarantee of the dependent rounding scheme
  - New conflict resolution heuristics

Concluding Statements

- Modelled the maximum facility location problem as an integer/linear program
- Presented a 0.19-approximation algorithm
- Discussed a dependent rounding scheme
- Future work:
  - Improving the 0.19-approximation algorithm's guarantee
  - Analysing the guarantee of the dependent rounding scheme
  - New conflict resolution heuristics

Questions?