Incremental Medians

Marek Chrobak  -  University of California, Riverside
Claire Kenyon  -  Brown University
John Noga  -  California State University, Northridge
Neal Young  -  University of California, Riverside
Given customers and facilities, choose k facilities to open. Minimize sum, over all customers c, of distance from c to nearest open facility.

\((3+\varepsilon)\)-approximation algorithm for metric case. [Araya et al, 2001]
Given customers and facilities, choose k facilities to open. Minimize sum, over all customers c, of distance from c to nearest open facility.

(3+\varepsilon)-approximation algorithm for metric case. [Araya et al, 2001]
Open facilities one at a time:

Minimize competitive ratio = $\max_{k=1}^{n} \frac{\text{cost}(F_k)}{\text{opt}_k}$

40-competitive algorithm for metric case. [Mettu & Plaxton, 2000]
Open facilities one at a time: $F_1$

Minimize competitive ratio $= \max_{k=1}^{n} \frac{\text{cost}(F_k)}{\text{opt}_k}$

40-competitive algorithm for metric case. [Mettu & Plaxton, 2000]
Open facilities one at a time: $F_1 \subset F_2$

Minimize competitive ratio = $\max_{k=1}^{n} \frac{\text{cost}(F_k)}{\text{opt}_k}$

40-competitive algorithm for metric case. [Mettu & Plaxton, 2000]
Open facilities one at a time: $F_1 \subseteq F_2 \subseteq F_3$

Minimize competitive ratio = $\max_{k=1}^{n} \frac{\text{cost}(F_k)}{\text{opt}_k}$

40-competitive algorithm for metric case. [Mettu & Plaxton, 2000]
Open facilities one at a time: $F_1 \subset F_2 \subset F_3 \subset F_4$

Minimize competitive ratio $= \max_{k=1}^{n} \frac{\text{cost}(F_k)}{\text{opt}_k}$

40-competitive algorithm for metric case. [Mettu & Plaxton, 2000]
Open facilities one at a time: \( F_1 \subset F_2 \subset F_3 \subset F_4 \subset F_5 \)

Minimize competitive ratio = \( \max_{k=1}^{n} \frac{\text{cost}(F_k)}{\text{opt}_k} \)

40-competitive algorithm for metric case. [Mettu & Plaxton, 2000]
Open facilities one at a time: $F_1 \subset F_2 \subset F_3 \subset F_4 \subset F_5 \subset F_6$

Minimize competitive ratio $= \max_{k=1}^{n} \frac{\text{cost}(F_k)}{\text{opt}_k}$

40-competitive algorithm for metric case. [Mettu & Plaxton, 2000]
Open facilities one at a time: \( F_1 \subseteq F_2 \subseteq F_3 \subseteq F_4 \subseteq F_5 \subseteq F_6 \cdots \)

Minimize competitive ratio = \( \max_{k=1}^{n} \frac{\text{cost}(F_k)}{\text{opt}_k} \)

40-competitive algorithm for metric case. [Mettu & Plaxton, 2000]
greedy algorithm not competitive

$F_1$

(greedy)

$F_2$

(greedy, cost $(n-1)/2$)

$OPT_2$

(cost 1)
lower bound of 2 for any deterministic algorithm
reverse greedy

1. Let $F_n = \text{all facilities}$

2. For $k = n, n-1, n-2, \ldots, 2$ do

3. Choose facility $f$ in $F_k$ to minimize cost($F_k - \{f\}$).

4. $F_{k-1} = F_k - \{f\}$.

**upper bound:** $2\log(n)$-competitive

**lower bound:** $\Omega(\log(n)/\log \log n)$ (won’t show today)
**projection lemma**

For any $F_k$ and any $h$ there exists $P_h \subseteq F_k$ of size $h$ such that $\text{cost}(P_h) \leq \text{cost}(F_k) + 2 \text{OPT}_h$.

**proof:**

1. **gather on $\text{OPT}_h$**
   - $\text{cost} = \text{OPT}_h$

2. **return in $h$ groups**
   - $\text{cost} = \text{OPT}_h$

3. **move groups to $F_k$**
   - $\text{cost} = \text{cost}(F_k)$
projection lemma

For any \( F_k \) and any \( h \) there exists \( P_h \subseteq F_k \) of size \( h \) such that \( \text{cost}(P_h) \leq \text{cost}(F_k) + 2 \text{OPT}_h \).

proof:

1. gather on \( \text{OPT}_h \)
   \[ \text{cost} = \text{OPT}_h \]

2. return in \( h \) groups
   \[ \text{cost} = \text{OPT}_h \]

3. move groups to \( F_k \)
   \[ \text{cost} = \text{cost}(F_k) \]
**Projection Lemma**

For any $F_k$ and any $h$ there exists $P_h \subseteq F_k$ of size $h$ such that $\text{cost}(P_h) \leq \text{cost}(F_k) + 2\ \text{OPT}_h$.

**Proof:**

1. Gather on $\text{OPT}_h$, $\text{cost} = \text{OPT}_h$
2. Return in $h$ groups, $\text{cost} = \text{OPT}_h$
3. Move groups to $F_k$, $\text{cost} = \text{cost}(F_k)$
**projection lemma**

For any $F_k$ and any $h$ there exists $P_h \subseteq F_k$ of size $h$ such that $\text{cost}(P_h) \leq \text{cost}(F_k) + 2 \text{OPT}_h$.

**proof:**

- 1. gather on $\text{OPT}_h$
  
  $\text{cost} = \text{OPT}_h$

- 2. return in $h$ groups
  
  $\text{cost} = \text{OPT}_h$

- 3. move groups to $F_k$
  
  $\text{cost} = \text{cost}(F_k)$
projection lemma

For any \( F_k \) and any \( h \) there exists \( P_h \subseteq F_k \) of size \( h \) such that
\[
\text{cost}(P_h) \leq \text{cost}(F_k) + 2 \text{OPT}_h.
\]

proof:

1. gather on \( \text{OPT}_h \) 
   \( \text{cost} = \text{OPT}_h \)

2. return in \( h \) groups
   \( \text{cost} = \text{OPT}_h \)

3. move groups to \( F_k \)
   \( \text{cost} = \text{cost}(F_k) \)
projection lemma

For any \( F_k \) and any \( h \) there exists \( P_h \subseteq F_k \) of size \( h \) such that
\[
\text{cost}(P_h) \leq \text{cost}(F_k) + 2 \text{OPT}_h.
\]

proof:

expected cost

1. gather on \( \text{OPT}_h \)
   \[
   \text{cost} = \text{OPT}_h
   \]

2. return in \( h \) groups
   \[
   \text{cost} = \text{OPT}_h
   \]

3. move groups to \( F_k \)
   \[
   \text{cost} = \text{cost}(F_k)
   \]
projection lemma

For any $F_k$ and any $h$ there exists $P_h \subseteq F_k$ of size $h$

such that $\text{cost}(P_h) \leq \text{cost}(F_k) + 2 \text{OPT}_h$.

proof:

expected cost

1. gather on $\text{OPT}_h$
   $\text{cost} = \text{OPT}_h$

2. return in $h$ groups
   $\text{cost} = \text{OPT}_h$

3. move groups to $F_k$
   $\text{cost} = \text{cost}(F_k)$
**projection lemma**

For any $F_k$ and any $h$ there exists $P_h \subseteq F_k$ of size $h$

such that $\text{cost}(P_h) \leq \text{cost}(F_k) + 2 \text{OPT}_h$.

**proof:**

1. gather on $\text{OPT}_h$
   
   cost = $\text{OPT}_h$

2. return in $h$ groups
   
   cost = $\text{OPT}_h$

3. move groups to $F_k$
   
   cost = $\text{cost}(F_k)$
**corollary**

*For any $F_k$ and any $h$ there exists $f$ in $F_k$ with*

\[
\text{cost}(F_k - \{ f \}) \leq \text{cost}(F_k) + 2 \text{OPT}_h / (k-h).
\]

**proof:**

Projection lemma $\Rightarrow$ removing *all* $k-h$ facilities in $F_k - P_h$ would increase cost by at most $2 \text{OPT}_h$...

So there must be *one* to remove that increases cost by at most $2 \text{OPT}_h$ over $k-h$. 


corollary: reverse greedy is $2 \, H_n$-competitive

1. Showed there exists $f$ in $F_k$ with

$$\text{cost}(F_k - \{f\}) \leq \text{cost}(F_k) + \frac{2 \text{OPT}_h}{k - h}.$$

2. Thus (taking $k = n, n - 1, n - 2, \ldots, h + 1$),

$$\text{cost}(F_h) \leq 2 \text{OPT}_h \left[ \frac{1}{n-h} + \frac{1}{n-1-h} + \cdots + \frac{1}{1} \right].$$
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

cost bound

F(20)

opt(20)
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

(cost bound)

opt(20)

F(20)

F(14)
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen $k$'s:

cost bound

$\text{opt}(20)$

$F(20)$

$F(14)$
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

cost bound

opt(20)

+ 2 opt(14)
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

\[ \text{cost bound} \]

\[ \text{opt}(20) \]

\[ + 2 \text{ opt}(14) \]
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

cost bound

opt(20)

+ 2 opt(14)
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

- cost bound
- opt(20)
- + 2 opt(14)
- + 2 opt(10)
better algorithm – “batch” reverse greedy

Project down at a few well-chosen k’s:

cost bound

\[ \text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) \]

\[ \text{cost bound} \]

\[ \text{e.g. } \text{cost}(\text{F}(6)) \leq \text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6) \]
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

cost bound

opt(20)
+ 2 opt(14)
+ 2 opt(10)

\[ \text{e.g. } \text{cost}(F(6)) \leq \text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6) \]
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen $k$’s:

\[
\text{cost bound} = \text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6)
\]

\[\text{e.g. } \text{cost}(F(6)) \leq \text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6)\]
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

\[ \text{cost bound} \]
\[ \text{opt}(20) \]
\[ + 2 \text{opt}(14) \]
\[ + 2 \text{opt}(10) \]
\[ + 2 \text{opt}(6) \]

e.g. \( \text{cost}(F(6)) \leq \text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6) \)
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

\[ \text{cost bound} \]

opt(20)

+ 2 opt(14)

+ 2 opt(10)

+ 2 opt(6)

e.g. \( \text{cost}(F(6)) \leq \text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6) \)
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen $k$'s:

e.g. $\text{cost}(F(6)) \leq \text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6)$
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen $k$'s:

\[ \text{cost bound} \]

\[ \text{opt}(20) \]

\[ + 2 \text{ opt}(14) \]

\[ + 2 \text{ opt}(10) \]

\[ + 2 \text{ opt}(6) \]

\[ + 2 \text{ opt}(3) \]

\[ \text{F}(1) \]

\[ \text{F}(3) \]

\[ \text{F}(6) \]

\[ \text{F}(10) \]

\[ \text{F}(14) \]

\[ \text{F}(20) \]

e.g. \( \text{cost}(\text{F}(6)) \leq \text{opt}(20) + 2 \text{ opt}(14) + 2 \text{ opt}(10) + 2 \text{ opt}(6) \)
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

\[
\text{cost bound} \\
\text{opt}(20) \\
+ 2 \text{opt}(14) \\
+ 2 \text{opt}(10) \\
+ 2 \text{opt}(6) \\
+ 2 \text{opt}(3)
\]

\[\text{e.g. } \text{cost}(F(6)) \leq \text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6)\]
better algorithm -- “batch” reverse greedy

Project down at a few well-chosen k’s:

e.g. $\text{cost}(F(6)) \leq \text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6)$
choose the k’s so opt costs double
choose the k’s so opt costs double

for \( k=9 \) we use \( F(6) \), of cost at most

\[
2 \text{opt}(6) + 2 \text{opt}(10) + 2 \text{opt}(14) + 2 \text{opt}(20) < 4 \text{opt}(9) + 2 \text{opt}(9) + \text{opt}(9) + \text{opt}(9)/2 < 8 \text{opt}(9)
\]
result (upper bound on competitive ratio)

<table>
<thead>
<tr>
<th></th>
<th>exponential time</th>
<th>polynomial time</th>
</tr>
</thead>
<tbody>
<tr>
<td>deterministic</td>
<td>8</td>
<td>$8(3+\varepsilon)$ = $24+\varepsilon$</td>
</tr>
<tr>
<td>randomized</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>
online bidding

instance: an unknown threshold $T \geq 1$.

to play: submit bids until threshold is exceeded.

competitive ratio: max over $T$ of $(\text{sum of bids submitted}) / T$

intuition

Consider possible bids $b \in [1, \infty)$

Have to choose larger and larger bids.

$T$ can be as small as last bid, so next bid can’t be too large.
But bids that are too close together give high aggregate cost.
choosing the k’s reduces to online bidding

e.g. \( k = 9 \Leftrightarrow \text{threshold } \operatorname{opt}(9) \)

Pay bids \( \operatorname{opt}(20), \operatorname{opt}(14), \operatorname{opt}(10), \operatorname{opt}(6) \).

Competitive ratio is (twice)

\[
\frac{\operatorname{opt}(20) + \operatorname{opt}(14) + \operatorname{opt}(10) + \operatorname{opt}(6)}{\operatorname{opt}(9)}
\]
### Doubling strategy
Submit bids 1, 2, 4, 8, 16, ...

### Doubling strategy is 4-competitive.
#### Proof:
Sum of bids is at most $1+2+4+8+\ldots+2T < 4T$.

This is the best possible competitive ratio.
Randomized online bidding

strategy: submit bids $e^x$, $e^{1+x}$, $e^{2+x}$, $e^{3+x}$, $e^{4+x}$, ...
where $x$ is chosen uniformly in $[0,1]$

Randomized strategy is $e$-competitive.

This is best possible for any randomized strategy.
lower bound for randomized online bidding

optimal strategy is solution to linear program:

\[ x(t, b) = \text{probability } b \text{ is first bid that is } t \text{ or larger} \]

\[ \beta = \text{competitive ratio} \]

\[
\begin{align*}
\text{minimize}_{\beta, x} & \quad \beta \quad \text{subject to} \\
\quad \beta - \frac{1}{n} \sum_{b=1}^{n} \frac{1}{T} \sum_{t=1}^{T} x(t, b) & \geq 0 \quad (\forall T \in [n]) \\
\frac{1}{n} \sum_{b=T}^{n} \sum_{t=1}^{T} x(t, b) & \geq 1 \quad (\forall T \in [n]) \\
x(t, b) & \geq 0 \quad (\forall t, b \in [n]).
\end{align*}
\]
lower bound follows from analytic solution to dual:

\[
\begin{align*}
\text{maximize}_{\mu, \pi} & \quad \sum_{T=1}^{n} \mu(T) \quad \text{subject to} \quad \sum_{T=1}^{n} \pi(T) \leq 1 \\
\sum_{T=t}^{b} \mu(T) - \sum_{T=t}^{n} \frac{b}{T} \pi(T) & \leq 0 \quad (\forall t, b \in [n]) \\
\mu(T), \pi(T) & \geq 0 \quad (\forall T \in [n]).
\end{align*}
\]

\[
\mu(T) = \begin{cases} 
\alpha/T & \text{if } U \leq T \leq U^2 \\
0 & \text{otherwise}
\end{cases}
\quad \text{and} \quad
\pi(T) = \begin{cases} 
1/T & \text{if } U \leq T \leq U^2 \log U \\
0 & \text{otherwise}.
\end{cases}
\]
### Result (Upper Bounds on Competitive Ratio)

<table>
<thead>
<tr>
<th></th>
<th>Exponential Time</th>
<th>Polynomial Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic</td>
<td>8</td>
<td>$8(3+\varepsilon)$ = $24+\varepsilon$</td>
</tr>
<tr>
<td>Randomized</td>
<td>$2e &lt; 5.44$</td>
<td>$2e(3+\varepsilon) &lt; 16.31$</td>
</tr>
</tbody>
</table>
thank you
projection lemma

For any $F_k$ and any $j$ there exists $F_j \subseteq F_k$ with

$$\text{cost}(F_j) \leq 2 \text{OPT}_j + \text{cost}(F_k).$$

proof:
For each point in $\text{OPT}_j$ take closest point in $F_k$.
Apply triangle inequality.

for any customer $c$

$$d(c,f_j) \leq d(c,\text{opt}_j) + d(\text{opt}_j,f_j)$$
$$\leq d(c,\text{opt}_j) + d(\text{opt}_j,f_k)$$
$$\leq d(c,\text{opt}_j) + d(\text{opt}_j,c) + d(c,f_k)$$
projection lemma

For any $F_k$ and any $j$
there exists $F_j \subseteq F_k$ with

$$\text{cost}(F_j) \leq 2 \text{OPT}_j + \text{cost}(F_k).$$

proof:
For each point in $\text{OPT}_j$
take closest point in $F_k$.
Apply triangle inequality.

for any customer $c$

$$d(c,f_j) \leq d(c,\text{opt}_j) + d(\text{opt}_j,f_j)$$
$$\leq d(c,\text{opt}_j) + d(\text{opt}_j,f_k)$$
$$\leq d(c,\text{opt}_j) + d(\text{opt}_j,c) + d(c,f_k)$$
## Choosing When to Project

<table>
<thead>
<tr>
<th>k</th>
<th>Project?</th>
<th>F(k)</th>
<th>Cost Bound</th>
<th>Versus</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td></td>
<td>F(20)</td>
<td>(\text{opt}(20))</td>
<td>(\text{opt}(20))</td>
</tr>
<tr>
<td>19</td>
<td>no</td>
<td>F(14)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14))</td>
<td>(\text{opt}(19))</td>
</tr>
<tr>
<td>18</td>
<td>no</td>
<td>F(14)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14))</td>
<td>(\text{opt}(18))</td>
</tr>
<tr>
<td>17</td>
<td>no</td>
<td>F(14)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14))</td>
<td>(\text{opt}(17))</td>
</tr>
<tr>
<td>16</td>
<td>no</td>
<td>F(14)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14))</td>
<td>(\text{opt}(16))</td>
</tr>
<tr>
<td>15</td>
<td>no</td>
<td>F(14)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14))</td>
<td>(\text{opt}(15))</td>
</tr>
<tr>
<td>14</td>
<td>yes</td>
<td>F(14)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14))</td>
<td>(\text{opt}(14))</td>
</tr>
<tr>
<td>13</td>
<td>no</td>
<td>F(10)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10))</td>
<td>(\text{opt}(13))</td>
</tr>
<tr>
<td>12</td>
<td>no</td>
<td>F(10)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10))</td>
<td>(\text{opt}(12))</td>
</tr>
<tr>
<td>11</td>
<td>no</td>
<td>F(10)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10))</td>
<td>(\text{opt}(11))</td>
</tr>
<tr>
<td>10</td>
<td>yes</td>
<td>F(10)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10))</td>
<td>(\text{opt}(10))</td>
</tr>
<tr>
<td>9</td>
<td>no</td>
<td>F(6)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6))</td>
<td>(\text{opt}(9))</td>
</tr>
<tr>
<td>8</td>
<td>no</td>
<td>F(6)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6))</td>
<td>(\text{opt}(8))</td>
</tr>
<tr>
<td>7</td>
<td>no</td>
<td>F(6)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6))</td>
<td>(\text{opt}(7))</td>
</tr>
<tr>
<td>6</td>
<td>yes</td>
<td>F(6)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6))</td>
<td>(\text{opt}(6))</td>
</tr>
<tr>
<td>5</td>
<td>no</td>
<td>F(3)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6) + 2 \text{opt}(3))</td>
<td>(\text{opt}(2))</td>
</tr>
<tr>
<td>4</td>
<td>no</td>
<td>F(3)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6) + 2 \text{opt}(3))</td>
<td>(\text{opt}(4))</td>
</tr>
<tr>
<td>3</td>
<td>yes</td>
<td>F(3)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6) + 2 \text{opt}(3))</td>
<td>(\text{opt}(3))</td>
</tr>
<tr>
<td>2</td>
<td>no</td>
<td>F(1)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6) + 2 \text{opt}(3) + 2 \text{opt}(1))</td>
<td>(\text{opt}(2))</td>
</tr>
<tr>
<td>1</td>
<td>yes</td>
<td>F(1)</td>
<td>(\text{opt}(20) + 2 \text{opt}(14) + 2 \text{opt}(10) + 2 \text{opt}(6) + 2 \text{opt}(3) + 2 \text{opt}(1))</td>
<td>(\text{opt}(1))</td>
</tr>
</tbody>
</table>