An exact approach for the capacitated p-center problem

Raphael Kramer[a]
Manuel Iori[a]
Thibaut Vidal[b]

[a]Università degli Studi di Modena e Reggio Emilia
[b]Pontifícia Universidade Católica do Rio de Janeiro

May 26th, 2016

Outline

Outline

Motivation

- Where to locate facilities?

Problem Definition — **p-center problem**

- Objective: Minimize the maximum distance from any customer to the nearest facility
- Applications: locating fire stations, police stations, hospitals, wireless routers, ...

Mathematical Formulation — **p-center problem**

- Input:
  - $C = \{1, 2, \ldots, n\}$ be the set of clients
  - $F = \{1, 2, \ldots, m\}$ be the set of potential sites or facilities
  - d_{ij} be the distance from client $j \in C$ to facility $i \in F$
  - p max. number of facilities to open
- Variables:
  - $x_{ij} = 1$ if customer j is covered by facility i; 0 if not
  - $y_i = 1$ if a facility is located in i; 0 if not

Mathematical Formulation — **p-center problem**

$$\text{(pCP) Minimize} \quad z \tag{1}$$

$$\text{s.t.} \quad \sum_{i \in F} x_{ij} = 1 \qquad j \in C, \tag{2}$$

$$x_{ij} \leq y_i \qquad i \in F, j \in C, \tag{3}$$

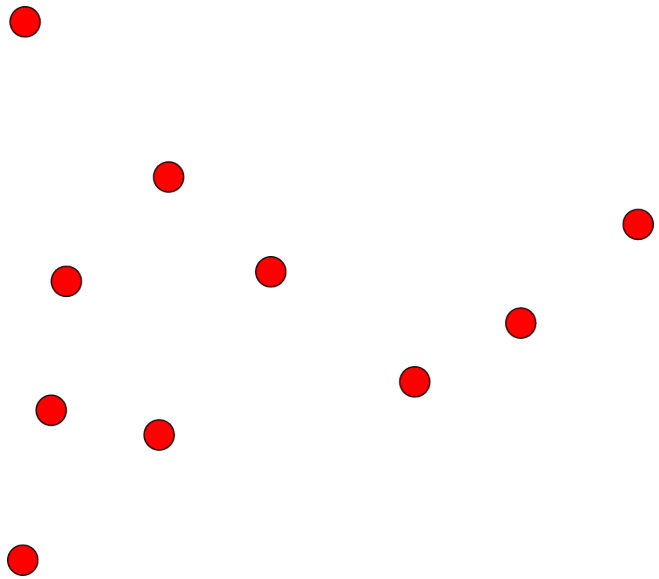
$$\sum_{i \in F} y_i \leq p, \tag{4}$$

$$z \geq \sum_{i \in F} d_{ij}\, x_{ij} \qquad j \in C, \tag{5}$$

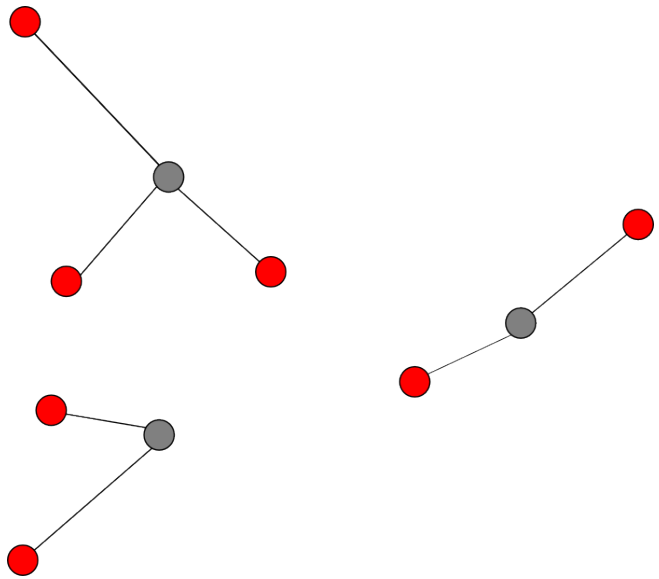
$$x_{ij} \in \{0, 1\} \qquad j \in C, i \in F, \tag{6}$$

$$y_i \in \{0, 1\} \qquad i \in F. \tag{7}$$

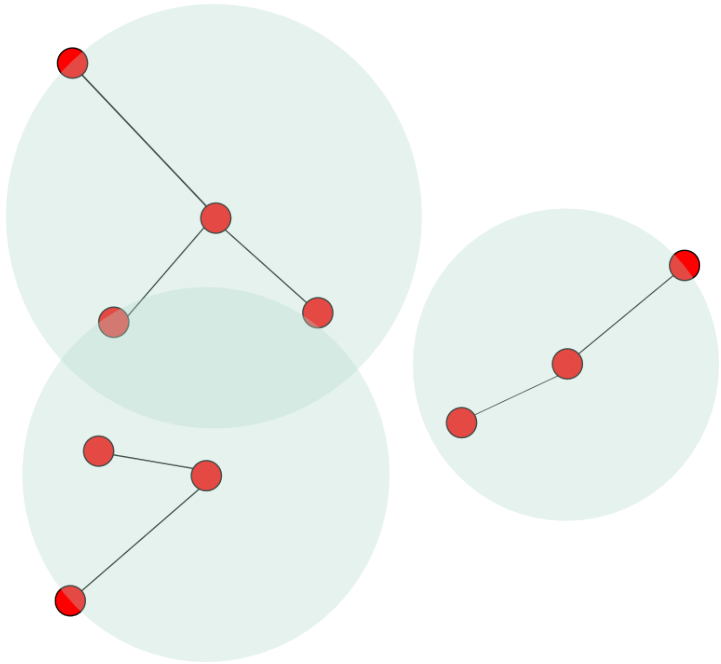
Example - 10 points, 3 facilities (wireless routers)



Example - 10 points, 3 facilities (wireless routers)



Example - 10 points, 3 facilities (wireless routers)



Capacitated p-center problem

- Additional inputs:
  - Q_i - capacity of facility $i \in F$
  - q_j - demand of client $j \in C$

Capacity Constraints:

$$\sum_{j \in C} q_j x_{ij} \leq Q_i y_i, \qquad i \in F$$

Mathematical Formulation — **cap. p-center problem**

$$
\begin{aligned}
\text{(CPCP) Minimize} \quad & z & & \\
\text{s.t.} \quad & \sum_{i \in F} x_{ij} = 1 & & j \in C, \\
& x_{ij} \leq y_i & & i \in F, j \in C, \\
& \sum_{i \in F} y_i \leq p, & & \\
& \sum_{j \in C} q_j x_{ij} \leq Q_i y_i & & i \in F, \qquad (8) \\
& z \geq \sum_{i \in F} d_{ij} x_{ij} & & j \in C, \\
& x_{ij} \in \{0,1\} & & j \in C, i \in F, \\
& y_i \in \{0,1\} & & i \in F.
\end{aligned}
$$

Outline

Relationship with covering problems

- Cap. Set Covering Problem
  - Input: Coverage radius **r** (demands, capacities, distances)
  - **Minimize** the number of facilities to cover all the clients
  - [Feasibility Problem: Setting $UB = p$]
- Cap. Maximal Covering Problem
  - Input: Coverage radius **r** (demands, capacities, distances, **p**)
  - **Maximize** the total demand covered by at most p facilities
  - [Feasibility Problem: Setting $LB = \sum_{j \in C} q_j$]

Relationship with covering problems

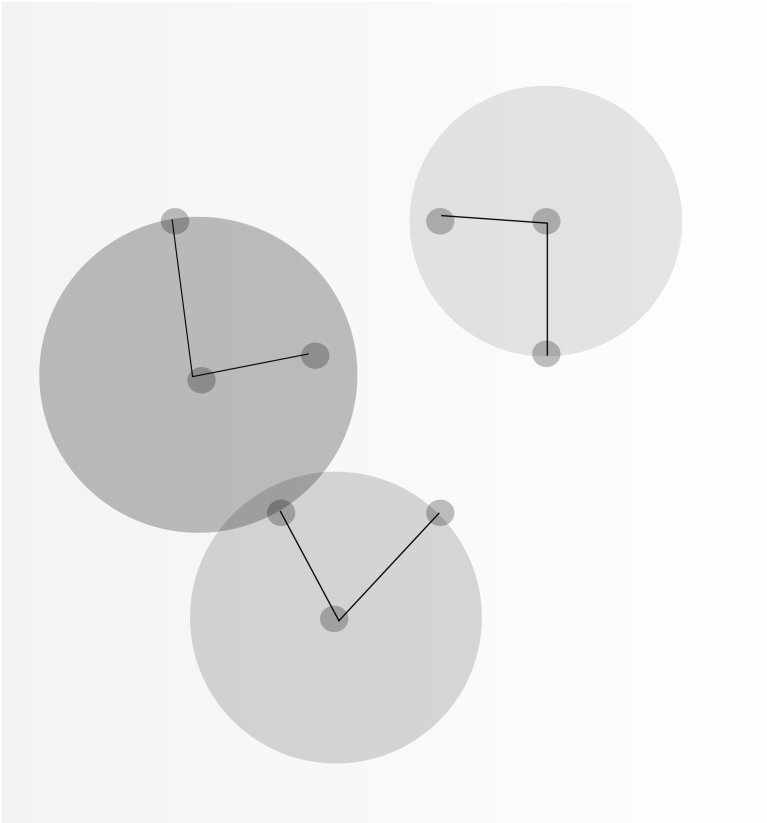


Figure : p-center solution

Relationship with covering problems

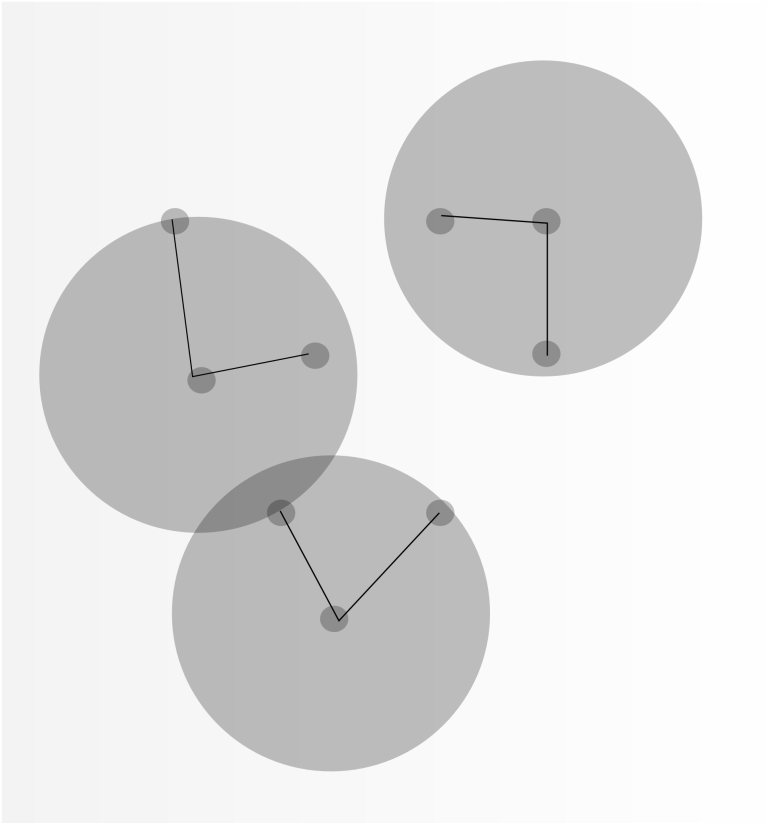


Figure : Set/Maximal Covering solution

Relationship with covering problems

- Minimize the maximum distance is equivalent to minimize the coverage radius

Objective:

Given n customers and m possible sites to locate at most p facilities, find the minimum coverage radius such that all customers can be covered

Binary search approach

The binary search algorithm for the pCP finds the minimum coverage radius (that covers all customers opening at most p facilities) within an sorted array $\mathbf{v}$ (containing all different values from the distance matrix). The optimal radius can be found after solving $\mathcal{O}(log|\mathbf{v}|)$ subproblems.

Example - Binary Search (minimization problem)

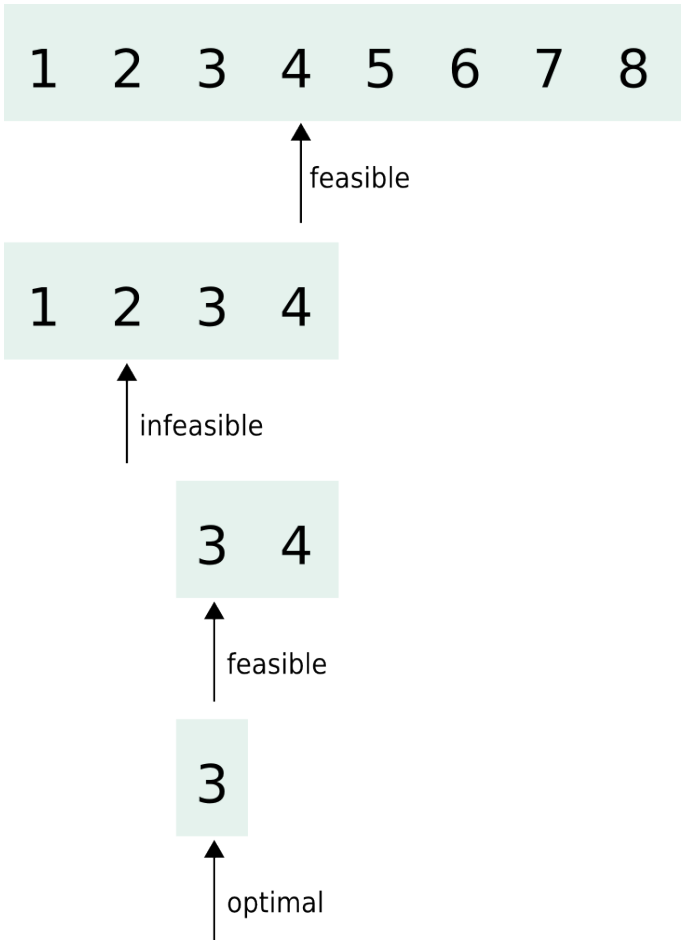
Distance Matrix

0	1	5	7	4
2	0	4	6	8
4	5	0	2	1
3	2	4	0	2
4	1	3	7	0

Coverage Radius

1	2	3	4	5	6	7	8

Example - Binary Search (minimization problem)



CPCP - Binary Search over Set Covering

- Capacitated Set Covering Formulation

$$\text{(CSCP-r) Minimize} \quad \sum_{i \in F} y_i \tag{9}$$

$$\text{s.t.} \quad \sum_{i \in F} a_{ij}^{r} y_i \geq 1 \qquad j \in C, \tag{10}$$

$$x_{ij} \leq y_i \qquad i \in F, j \in C, \tag{11}$$

$$\sum_{j \in C} q_j x_{ij} \leq Q_i y_i \qquad i \in F, \tag{12}$$

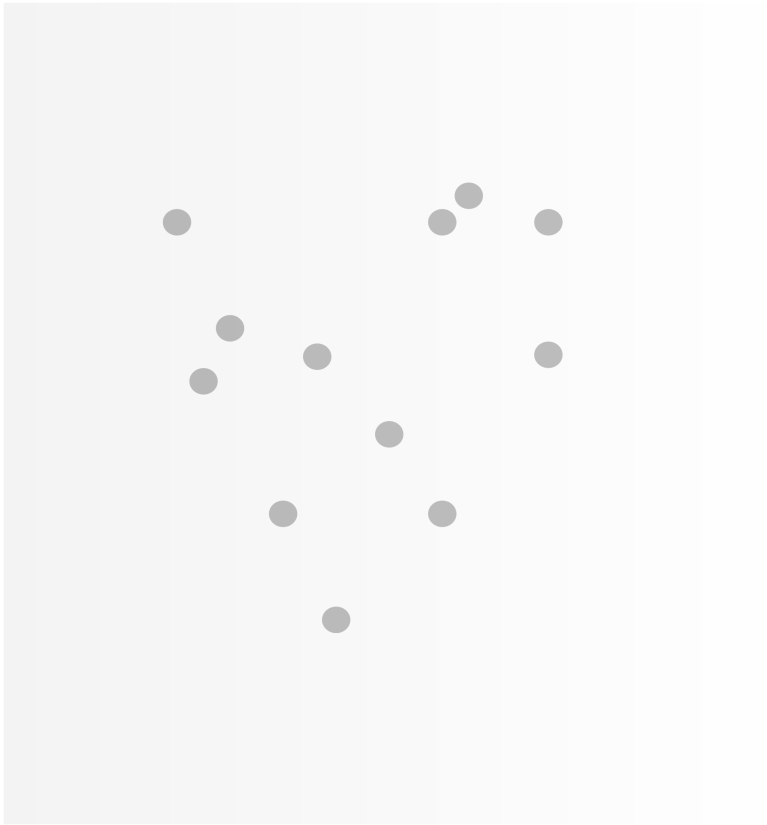
$$x_{ij} \in \{0,1\} \qquad i \in F, j \in C, \tag{13}$$

$$y_i \in \{0,1\} \qquad i \in F. \tag{14}$$

where, $a_{ij}^{r} = 1$, if $d_{ij} \leq r$; 0, if not.

- If the solution value is lower or equal to p, r is an upper bound to CPCP. Otherwise, $r + 1$ is a lower bound to CPCP.

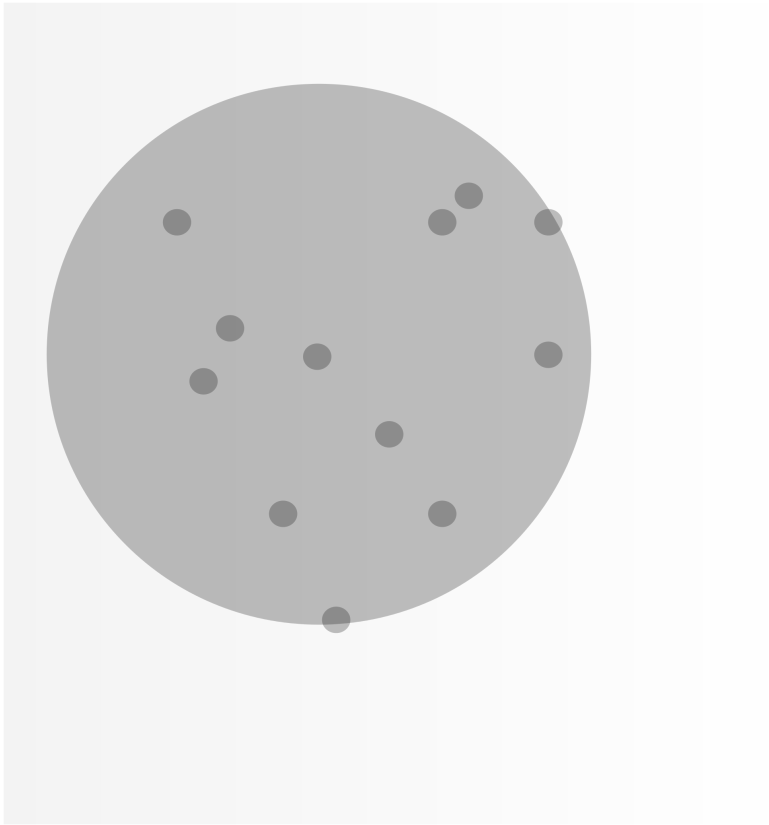
Example: Binary Search over Set Covering Subproblems



Instance: $n = 12$; $p = 3$

1 - 4 - 5 - 6 - 8 - 9 - 12 - 14 - 15 - 16 - 18 - 19 - 20 - 21 - 22 - 23 - 24 - 25 - 27 - 28 - 30

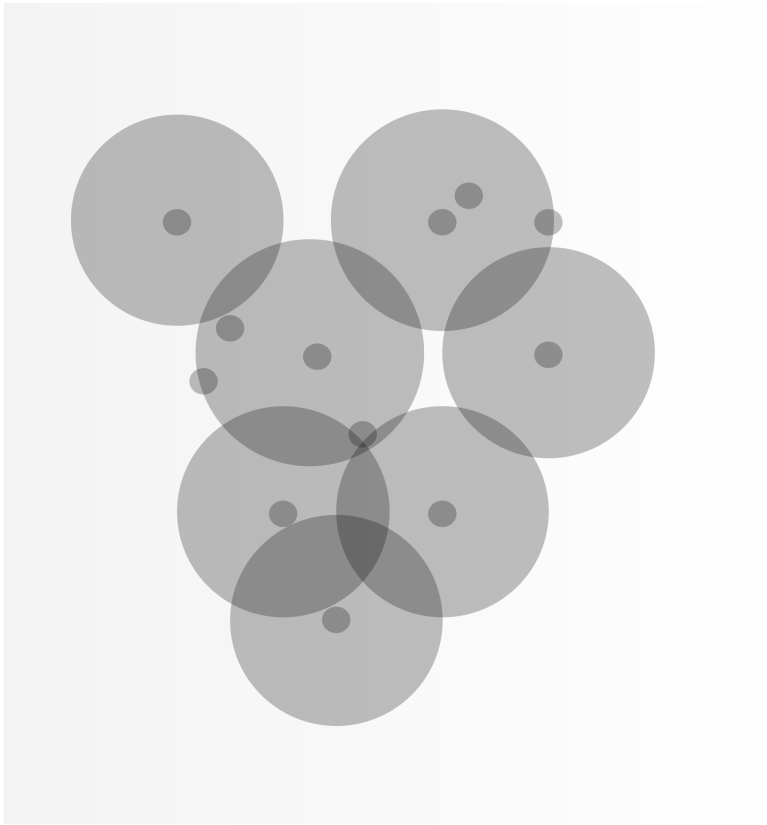
Example: Binary Search over Set Covering Subproblems



$r = 18,\ sol = 1$ (Feasible)

1 - 4 - 5 - 6 - 8 - 9 - 12 - 14 - 15 - 16 - 18 - 19 - 20 - 21 - 22 - 23 - 24 - 25 - 27 - 28 - 30

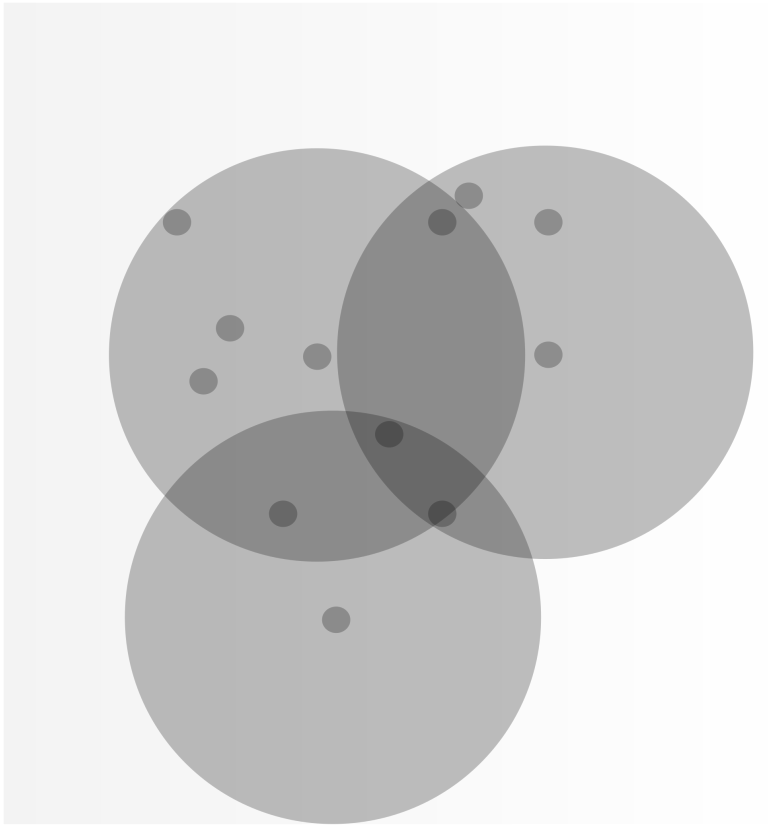
Example: Binary Search over Set Covering Subproblems



$r = 8,\ sol = 7$ (Infeasible)

1 - 4 - 5 - 6 - 8 - 9 - 12 - 14 - 15 - 16 - 18 - 19 - 20 - 21 - 22 - 23 - 24 - 25 - 27 - 28 - 30

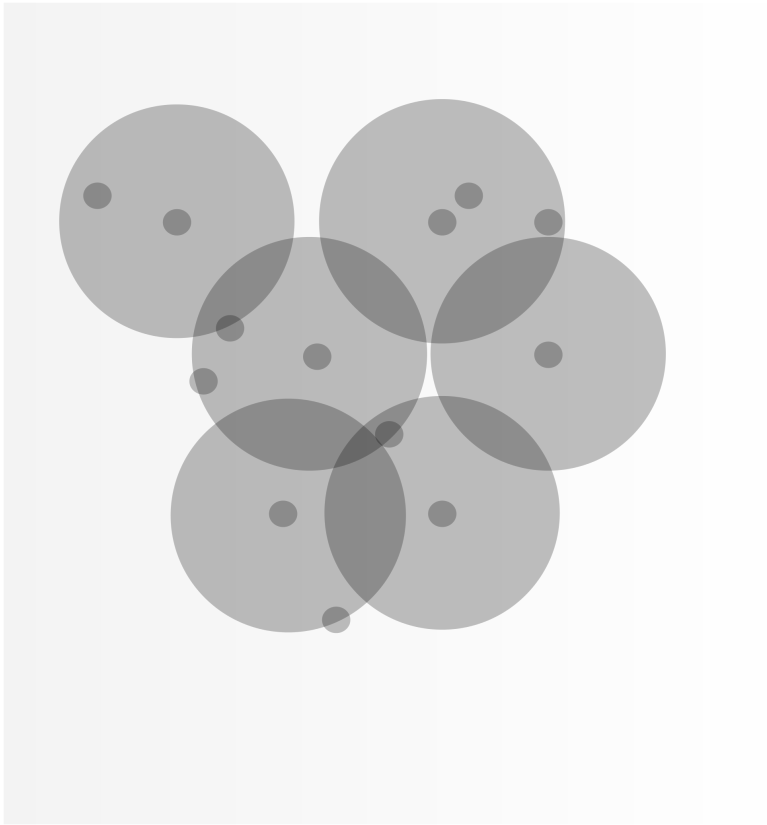
Example: Binary Search over Set Covering Subproblems



$r = 14,\ sol = 3$ (Feasible)

1 - 4 - 5 - 6 - 8 - 9 - 12 - 14 - 15 - 16 - 18 - 19 - 20 - 21 - 22 - 23 - 24 - 25 - 27 - 28 - 30

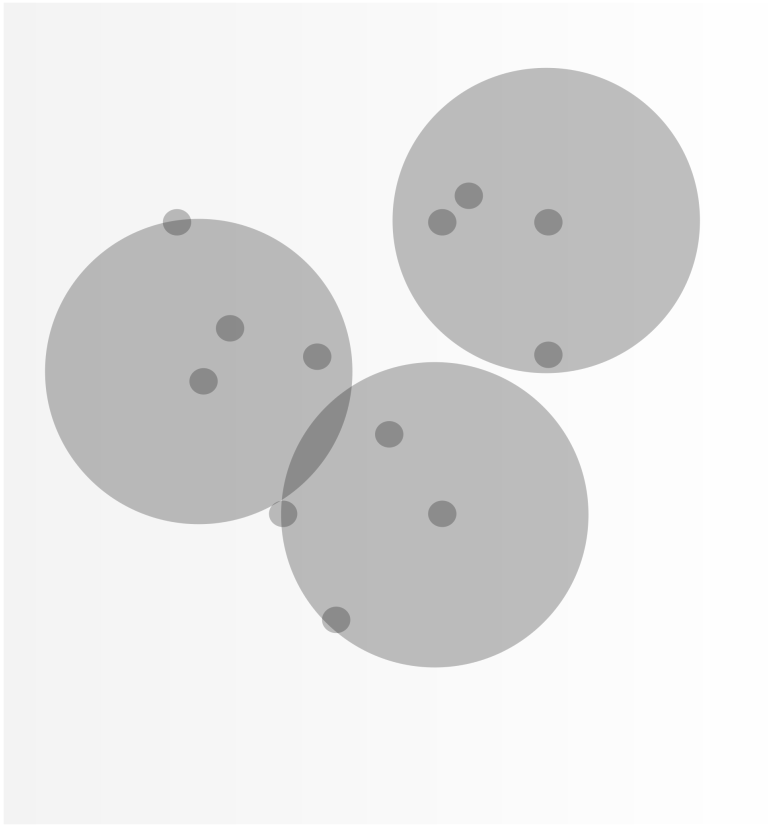
Example: Binary Search over Set Covering Subproblems



$r = 9,\ sol = 6$ (Infeasible)

1 - 4 - 5 - 6 - 8 - 9 - 12 - 14 - 15 - 16 - 18 - 19 - 20 - 21 - 22 - 23 - 24 - 25 - 27 - 28 - 30

Example: Binary Search over Set Covering Subproblems



$r = 12,\ sol = 3$ (Optimal)

1 - 4 - 5 - 6 - 8 - 9 - 12 - 14 - 15 - 16 - 18 - 19 - 20 - 21 - 22 - 23 - 24 - 25 - 27 - 28 - 30

Insights

- The efficiency of the binary search algorithm resides on how efficiently the subproblems are solved.
- Depending on the radius value some subproblems can be very difficult to be solved.
- Given the difficulty for solving some subproblems, the use of special "tricks" are important to improve the efficiency of the algorithm.
  - valid inequalities, domination rules, etc.

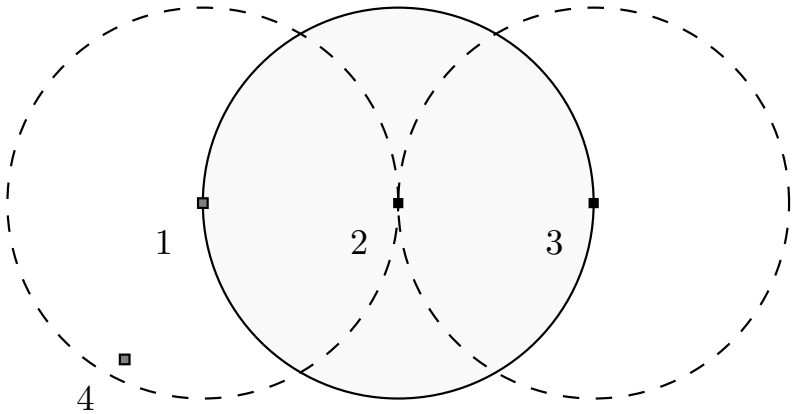
Outline

Additional constraints

- Additional constraint sets:
  - Domination rules
  - Break symmetries
  - Improve lower bound

Additional constraint set 1: Domination rule

- $C(i)$ - Set of clients covered by facility $i \in F$
- If $C(d) \subseteq C(i)$, then i dominates d
- $y_d \leq y_i$
  - If $\sum_{j \in C(i)} q_j \leq Q_i$, then $y_d = 0$

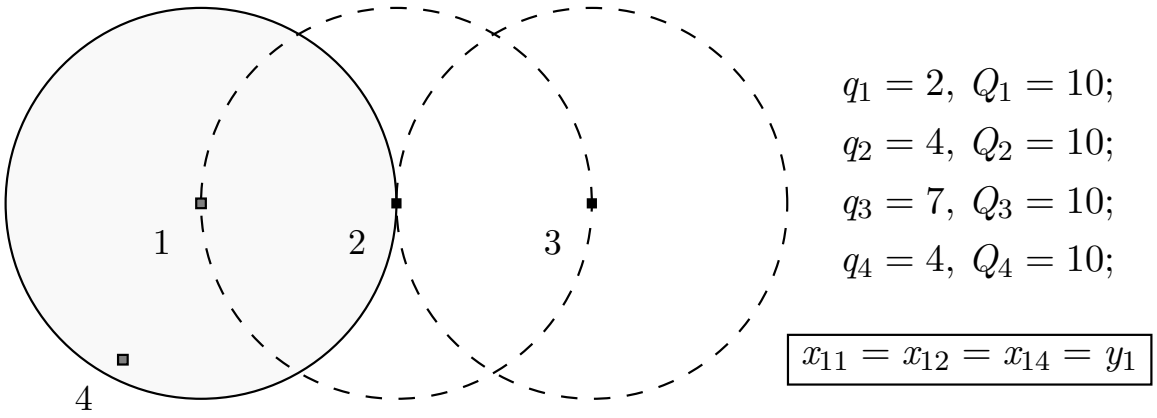


$C(1) = \{1, 2, 4\}$
$C(2) = \{1, 2, 3\}$
$C(3) = \{2, 3\}$

$\boxed{y_3 \leq y_2}$

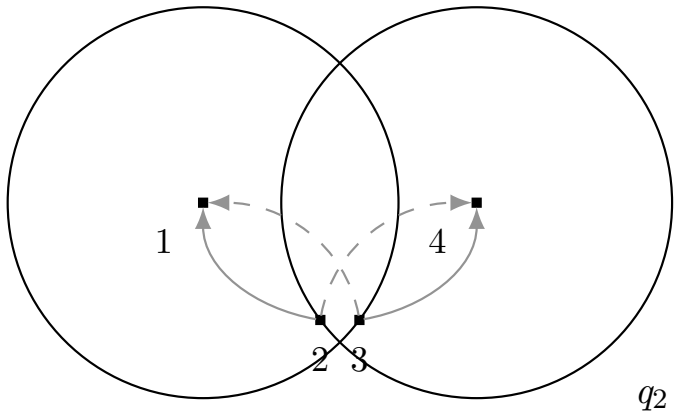
Additional constraint set 2: Forcing service

- If $\sum_{j \in C(i)} q_j \leq Q_i$, then
  - $x_{ij} = y_i, \quad \forall j \in C(i)$



Additional constraint set 3: Breaking symmetries

- 2 facilities (i_1 and i_2) and 2 clients (j_1 and j_2)
- Conditions:
  - $q_{j_1} = q_{j_2}$
  - $d_{i_1,j_1} \leq r$ and $d_{i_2,j_1} \leq r$ and $d_{i_2,j_1} \leq r$ and $d_{i_2,j_2} \leq r$
- If $d_{i_1,j_1} < d_{i_2,j_1}$ and $d_{i_2,j_2} < d_{i_i,j_2}$, then
  - $x_{i_1,j_2} + x_{i_2,j_1} \leq 1$



$q_2 = q_3$

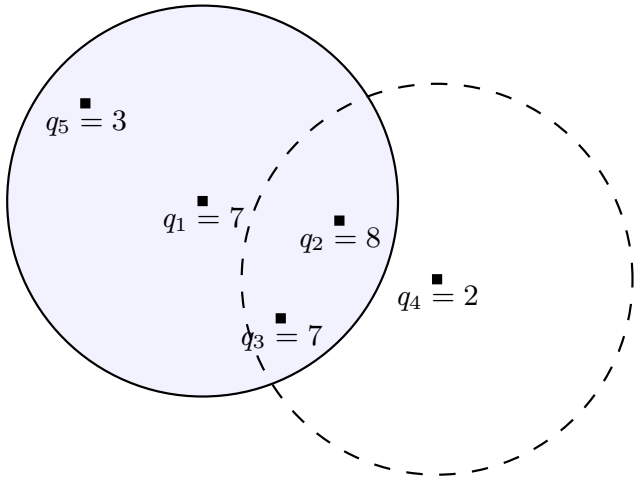
$\boxed{x_{13} + x_{42} \leq 1}$

Additional constraint set 4: Surplus Capacity Ineq.

- Given a facility $i \in F$
- If $(\sum_{j \in C(i)} q_j) > Q_i$, let
- $\bar{S}_i \subset C(i)$, such that $\sum_{j \in \bar{S}_i} q_j \geq (\sum_{j \in C(i)} q_j) - Q_i$
- Then,
  - $$\sum_{j \in C(i): j \notin \bar{S}_i} x_{ij} \geq (|C(i)| - |\bar{S}_i|) y_i - \sum_{j \in \bar{S}_i} M x_{ij}$$
  - $M = |C(i)| - |\bar{S}_i|$
  - $|\bar{S}_i| = \{1, 2\}$ (parameter)

Additional constraint set 5: Capacity-Cut Constraints

- $F(j)$ - Set of facilities covering client $j \in C$
- $S_i \subseteq C(i)$, such that $\sum_{j \in S_i} q_j > Q$
- $F(S_i) = \bigcup_{j \in S(i)} F(j)$
  - $\sum_{i \in F(S_i)} y_i \geq \left\lceil \frac{\sum_{j \in S_i} q_j}{Q} \right\rceil$
- $|S_i| = \{1, 2\}$ (parameter)



$S_1 = \{1, 2, 3\}$

$F(S_1) = \{1, 2, 3, 4\}$

$$\boxed{y_1 + y_2 + y_3 + y_4 \geq 3}$$

Subset sum lifting

- Based on the subset sum lifting of Boschetti and Mingozzi (2003) for the Bin Packing Problem
- Subset Sum Problem (SSP)
  - Special case of the Knapsack Problem, where profits are equal to weights
  - Objective: Maximize the total weight packed
- Solve a SSP for every item j, forcing j to be selected
  - Lifting: Remaining capacity is added to the weight of j
  - Heuristic: Sort the items, considering the weights already lifted

Disjunctive based subset sum inequalities

- Consider the disjunction: either $k \in C$ is served or not by $i \in F$
  - $v(SSP^0_{ik})$ - Objval if client k is not served by facility i
  - $v(SSP^1_{ik})$ - Objval if client k is served by facility i

$$\sum_{j \in C(i): j \neq k} q_j x_{ij} \leq v(SSP^0_{ik}) y_i \quad (\text{if } x_{ik} = 0),$$

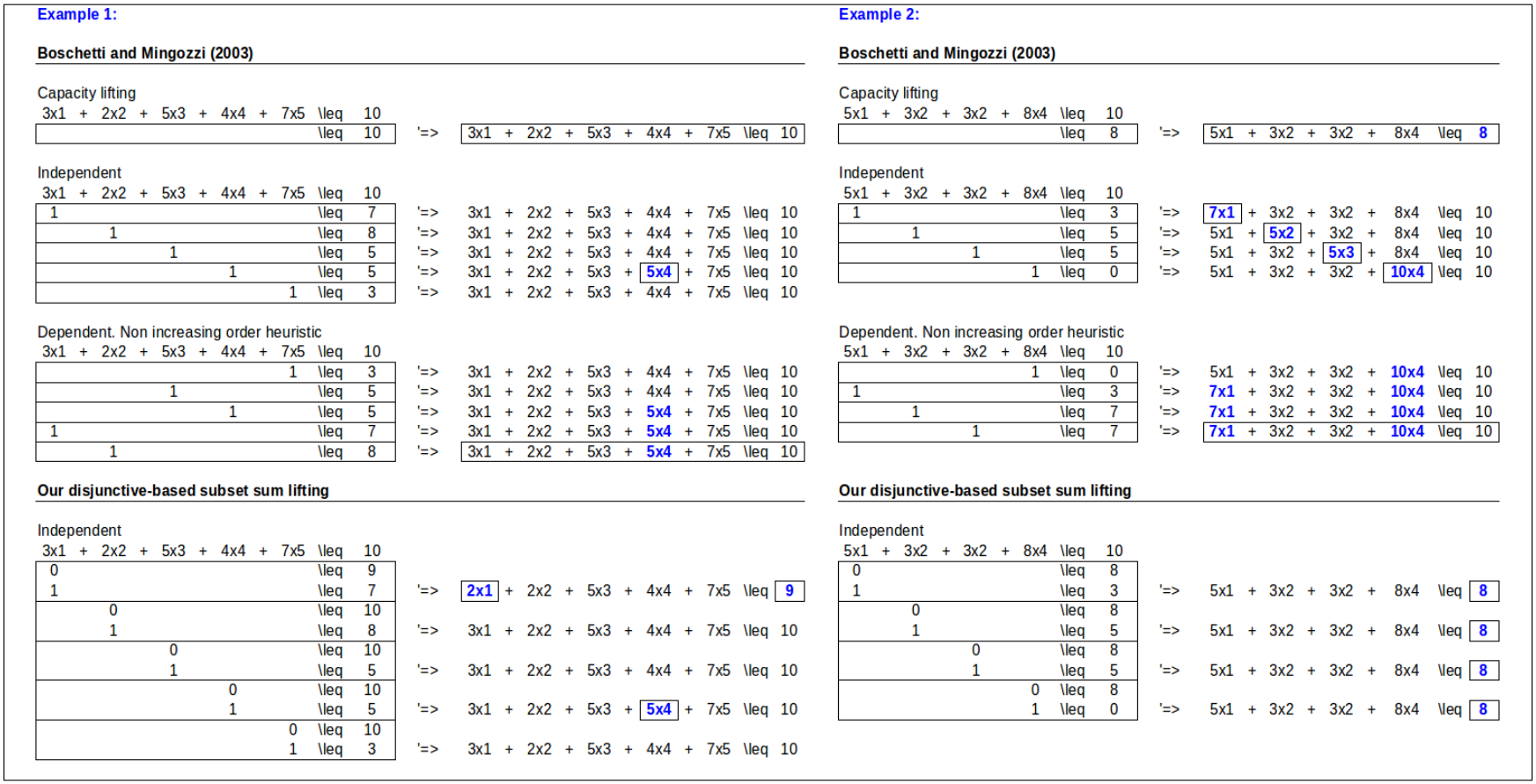
$$\sum_{j \in C(i): j \neq k} q_j x_{ij} \leq v(SSP^1_{ik}) y_i \quad (\text{if } x_{ik} = 1).$$

Hence,

$$\sum_{j \in C(i): j \neq k} q_j x_{ij} \leq v(SSP^0_{ik}) y_i - (v(SSP^0_{ik}) - v(SSP^1_{ik})) x_{ik}.$$

- $q_j = (v(SSP^0_{ik}) - v(SSP^1_{ik}))$
- $Q_i = v(SSP^0_{ik})$

Example SS Lifting: BM03 x Our



Example 1:

Boschetti and Mingozzi (2003)

Capacity lifting

3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10		3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
									\leq	10	=>	3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10

Independent

3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10		3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
1									\leq	7	=>	3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
		1							\leq	8	=>	3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
				1					\leq	5	=>	3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
						1			\leq	5	=>	3x1	+	2x2	+	5x3	+	**5x4**	+	7x5	\leq	10
								1	\leq	3	=>	3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10

Dependent. Non increasing order heuristic

3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10		3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
								1	\leq	3	=>	3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
				1					\leq	5	=>	3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
						1			\leq	5	=>	3x1	+	2x2	+	5x3	+	**5x4**	+	7x5	\leq	10
1									\leq	7	=>	3x1	+	2x2	+	5x3	+	**5x4**	+	7x5	\leq	10
		1							\leq	8	=>	3x1	+	2x2	+	5x3	+	**5x4**	+	7x5	\leq	10

Our disjunctive-based subset sum lifting

Independent

3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10		3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
0									\leq	9												
1									\leq	7	=>	**2x1**	+	2x2	+	5x3	+	4x4	+	7x5	\leq	9
		0							\leq	10												
		1							\leq	8	=>	3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
				0					\leq	10												
				1					\leq	5	=>	3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10
						0			\leq	10												
						1			\leq	5	=>	3x1	+	2x2	+	5x3	+	**5x4**	+	7x5	\leq	10
								0	\leq	10												
								1	\leq	3	=>	3x1	+	2x2	+	5x3	+	4x4	+	7x5	\leq	10

Example 2:

Boschetti and Mingozzi (2003)

Capacity lifting

5x1	+	3x2	+	3x2	+	8x4	\leq	10		5x1	+	3x2	+	3x2	+	8x4	\leq	10
							\leq	8	=>	5x1	+	3x2	+	3x2	+	8x4	\leq	**8**

Independent

5x1	+	3x2	+	3x2	+	8x4	\leq	10		5x1	+	3x2	+	3x2	+	8x4	\leq	10
1							\leq	3	=>	**7x1**	+	3x2	+	3x2	+	8x4	\leq	10
		1					\leq	5	=>	5x1	+	**5x2**	+	3x2	+	8x4	\leq	10
				1			\leq	5	=>	5x1	+	3x2	+	**5x3**	+	8x4	\leq	10
						1	\leq	0	=>	5x1	+	3x2	+	3x2	+	**10x4**	\leq	10

Dependent. Non increasing order heuristic

5x1	+	3x2	+	3x2	+	8x4	\leq	10		5x1	+	3x2	+	3x2	+	8x4	\leq	10
						1	\leq	0	=>	5x1	+	3x2	+	3x2	+	**10x4**	\leq	10
1							\leq	3	=>	**7x1**	+	3x2	+	3x2	+	**10x4**	\leq	10
		1					\leq	7	=>	**7x1**	+	3x2	+	3x2	+	**10x4**	\leq	10
				1			\leq	7	=>	**7x1**	+	3x2	+	3x2	+	**10x4**	\leq	10

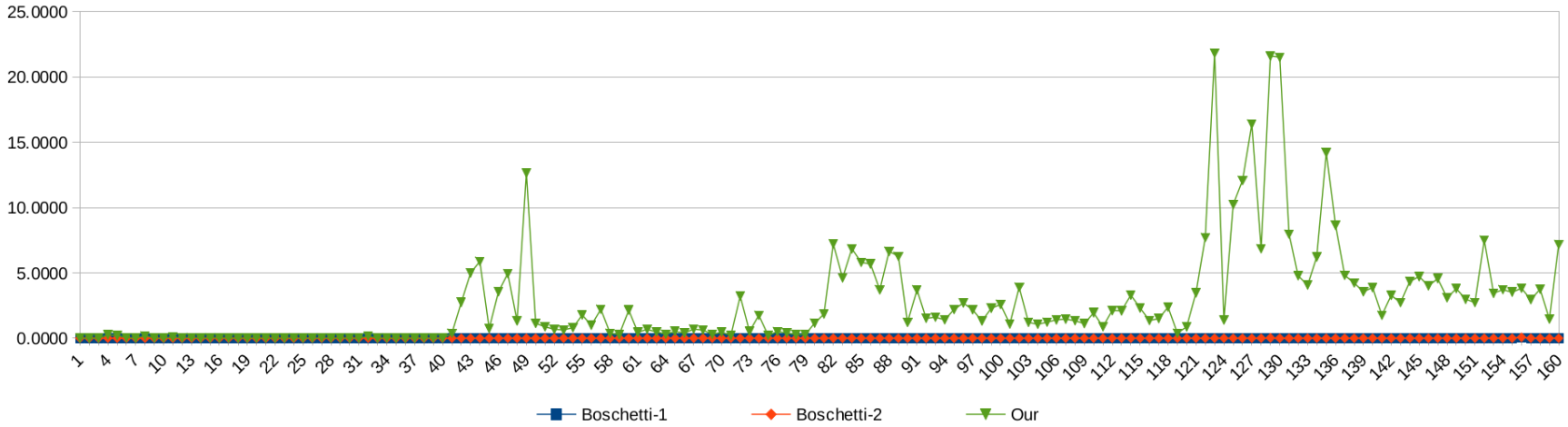
Our disjunctive-based subset sum lifting

Independent

5x1	+	3x2	+	3x2	+	8x4	\leq	10		5x1	+	3x2	+	3x2	+	8x4	\leq	10
0							\leq	8										
1							\leq	3	=>	5x1	+	3x2	+	3x2	+	8x4	\leq	**8**
		0					\leq	8										
		1					\leq	5	=>	5x1	+	3x2	+	3x2	+	8x4	\leq	**8**
				0			\leq	8										
				1			\leq	5	=>	5x1	+	3x2	+	3x2	+	8x4	\leq	**8**
						0	\leq	8										
						1	\leq	0	=>	5x1	+	3x2	+	3x2	+	8x4	\leq	**8**

Lower bound comparison SS Lifting: BM03 x Our

- CPCP Instances: $UB = opt + 2$
- Boschetti-1: Original capacities, lift the demands/weights
- Boschetti-2: Lift the capacities, lift the demands/weights
- Our: Disjunctive based subset sum inequalities
- Gap(%): $100\left(\frac{OrigLB - SSLB}{OrigLB}\right)$



LB Evaluation

Table 1. Average deviation to original lower bound

Instance	LBs Radius = r^*-1				LBs Radius = r^*			
	Orig	**CC1**	**CC2**	**SS**	**Orig**	**CC1**	**CC2**	**SS**
p550	5.28	0.00	0.00	0.00	4.82	0.00	0.00	0.00
p10100	10.10	0.00	0.00	0.00	9.57	0.00	0.00	0.00
p15150	15.67	0.00	0.00	0.01	14.25	0.00	0.00	0.01
p20200	20.39	0.00	0.00	0.01	18.87	0.00	0.00	0.01
p1250	11.77	0.00	0.09	0.17	11.27	0.00	0.05	0.17
p25100	24.20	0.00	0.20	0.26	23.04	0.00	0.00	0.15
p37150	36.29	0.00	0.27	0.53	33.73	0.00	0.07	0.39
p50200	48.85	0.00	0.74	0.80	46.17	0.00	0.47	0.51
p1650	15.51	0.06	0.29	0.28	15.00	0.01	0.04	0.21
p33100	31.43	0.06	0.55	0.65	30.12	0.00	0.17	0.40
p50150	47.67	0.23	1.39	1.54	45.10	0.10	0.70	0.96
p66200	63.50	0.58	2.18	2.34	59.82	0.19	0.87	1.15
p2050	18.58	0.10	0.33	0.79	18.41	0.03	0.05	0.52
p40100	36.87	0.47	0.81	1.64	36.18	0.18	0.41	0.98
p60150	55.30	0.69	1.33	2.40	53.15	0.31	0.56	1.45
p80200	74.38	0.82	2.05	3.36	71.90	0.44	0.71	1.78
Average		**0.19**	**0.64**	**0.92**		**0.08**	**0.26**	**0.54**

Performance Evaluation

Table 2. Best deviation to original lower bound

Instance	LBs Radius = r^*-1				LBs Radius = r^*			
	Orig	**CC1**	**CC2**	**SS**	**Orig**	**CC1**	**CC2**	**SS**
p550	5.28	0.00	0.00	0.01	4.82	0.00	0.00	0.01
p10100	10.10	0.00	0.00	0.01	9.57	0.00	0.00	0.02
p15150	15.67	0.00	0.00	0.02	14.25	0.00	0.00	0.02
p20200	20.39	0.00	0.00	0.03	18.87	0.00	0.00	0.03
p1250	11.77	0.00	0.40	0.57	11.27	0.00	0.00	0.42
p25100	24.20	0.00	0.80	0.79	23.04	0.00	0.05	0.42
p37150	36.29	0.00	0.85	0.93	33.73	0.00	0.27	0.60
p50200	48.85	0.00	1.70	1.43	46.17	0.00	1.15	0.94
p1650	15.51	0.53	1.70	1.43	15.00	0.00	1.15	0.94
p33100	31.43	0.53	1.70	1.43	30.12	0.01	1.15	0.94
p50150	47.67	0.89	2.65	2.18	45.10	0.26	1.86	1.66
p66200	63.50	1.84	5.15	3.74	59.82	0.70	2.15	2.04
p2050	18.58	1.84	5.15	3.74	18.41	0.70	2.15	2.04
p40100	36.87	1.84	5.15	3.74	36.18	0.70	2.15	2.04
p60150	55.30	3.69	5.15	4.16	53.15	0.99	2.15	2.76
p80200	74.38	3.69	5.15	5.77	71.90	1.73	2.15	3.05
Average		**0.93**	**2.22**	**1.87**		**0.32**	**1.02**	**1.12**

Outline

Computational Environment

- C++ programming language
- Intel Core i5 2.30GHz, 4 GB RAM
- Linux Mint 17.2 64-bit
- Gurobi Optimizer 6.5.1
- CPLEX Optimizer 12.5.1

Instances

- ORLIB
  - $|C| = |F| = \{50, 100, 150, 200\}$
  - $p = \left\{ \frac{|C|}{10}, \frac{|C|}{4}, \frac{|C|}{3}, \frac{|C|}{2.5} \right\}$
  - 160 instances in total

Parameters

- One thread
- Time Limit = 600 seconds

State-of-the-art results

- Scaparra et al. (Networks, 2004):
  - Heuristic method
- Össoy & Pınar (C&OR, 2006):
  - Cap. Set Partitioning
- Albareda-Sambola et. al. (EJOR, 2010):
  - Lagrangian Relaxation for CSCP, and CMCP
- Quevedo-Orozco & Ríos-Mercado (C&OR, 2015):
  - Heuristic method
- Summary: 13 open instances

Computational results

ORLIB *160 instances*	**Original Formulation CPCP**	**Subproblems** ($r = 1, 2, 3, \ldots, r^*$)					
		CPLEX			**GUROBI**		
		SC-All Ineq	**SP-Orig**	**SC-Orig**	**SC-All Ineq**	**SP-Orig**	**SC-Orig**
Non-optimal	35	**9**	13	13	**0**	5	6
Avg. Open Nodes	49,497.39	**4,373.14**	7,288.42	7,183.78	**807.29**	2,585.61	2,071.55
Avg. Time (s)	192.68	**57.46**	66.91	67.36	**15.84**	32.83	34.16

Computational results

Table 3. Gap to the optimal solution.

Instance			CPCP	Literature				Our-All Ineq.	
n	p	#		SPS	OP	ADF	QR	CPLEX	GUROBI
50	5	10	0	2.03	0	0	0	0	0
100	10	10	0	8.79	0	0	1.41	0	0
150	15	10	0	16.25	0	0	3.77	0	0
200	20	10	8.05	23.40	0	0.71	5.72	0	0
50	12	10	0	5.84	0	0.00	7.57	0	0
100	25	10	2.67	15.63	0.77	3.85	12.69	-1.38	0
150	37	10	0	23.41	1.00	0.91	15.51	-0.83	0
200	50	10	16.33	27.22	4.72	14.22	36.89	0	0
50	16	10	0	3.02	0	0	10.27	0	0
100	33	10	0.77	12.96	1.00	2.51	12.42	0	0
150	50	10	2	19.49	1.00	4.02	33.53	0	0
200	66	10	7.94	32.47	1.43	10.56	62.53	-1.11	0
50	20	10	0	3.89	0	2.00	30.62	0	0
100	40	10	1.68	11.62	0	4.45	61.49	0	0
150	60	10	1.11	19.74	1.11	4.79	78.71	-0.71	0
200	80	10	7.82	35.00	6.61	11.75	64.58	-1.11	0
Average			3.02	16.30	1.10	3.74	27.36	-0.32	0.00

Results obtained from:

Quevedo-Orozco, D. R.; Rios-Mercado, R. Z. 2015. "Improving the quality of heuristic solutions for the capacitated vertex p-center problem through iterated greedy local search with variable neighborhood descent." Computers & Operations Research 62 (1): 133-144.

Computational results

Table 4. Gap to the optimal solution. Updated.

Instance			CPCP	Literature				Our-All Ineq.	
n	p	#		SPS	OP	ADF	QR	CPLEX	GUROBI
50	5	10	0	2.03	0	0	0	0	0
100	10	10	0	8.79	0	0	1.41	0	0
150	15	10	0	16.25	0	0	3.77	0	0
200	20	10	8.05	23.40	0	0.71	5.72	0	0
50	12	10	0	5.84	0	0.00	7.57	0	0
100	25	10	2.67	15.63	0	2.87	11.51	-1.38	0
150	37	10	0	22.35	0	0	14.45	-0.83	0
200	50	10	16.33	27.22	3.22	12.78	34.44	0	0
50	16	10	0	3.02	0	0	10.27	0	0
100	33	10	0.77	11.87	0	1.60	11.43	0	0
150	50	10	2	19.49	0	3.11	32.24	0	0
200	66	10	7.94	30.38	0	9.31	60.86	-1.11	0
50	20	10	0	3.89	0	2.00	30.62	0	0
100	40	10	1.68	11.62	0	4.45	61.49	0	0
150	60	10	1.11	19.74	0	4.02	76.30	-0.71	0
200	80	10	7.82	31.67	2.73	7.98	59.72	-1.11	0
Average			3.02	15.82	0.37	3.05	26.36	-0.32	0.00

Results obtained from:

Quevedo-Orozco, D. R.; Rios-Mercado, R. Z. 2015. "Improving the quality of heuristic solutions for the capacitated vertex p-center problem through iterated greedy local search with variable neighborhood descent." Computers & Operations Research 62 (1): 133-144.

Outline

Conclude remarks

- CSCP + valid inequalities
- Improvement of subset sum lifting
- All instances solved to proven optimality
- Future works:
  - Integrate the proposed inequalities in an arc-flow formulation
  - Aggregation and disaggregation rules

Conclusion

Thank you!

An exact approach for the capacitated p-center problem

Raphael Kramer[a]
Manuel Iori[a]
Thibaut Vidal[b]

[a]Università degli Studi di Modena e Reggio Emilia
[b]Pontifícia Universidade Católica do Rio de Janeiro

May 26th, 2016