Approximation Algorithms
A Survey

David P. Williamson
IBM Watson

Discrete Optimization '99
July 27, 1999

Approximation Algorithms Since 1977

David P. Williamson
IBM Watson

Discrete Optimization '99

July 27, 1999

Recent Techniques in Approximation Algorithms

David P. Williamson
IBM Watson

Discrete Optimization '99

July 27, 1999

Three Recent Techniques in Approximation Algorithms

David P. Williamson
IBM Watson

Discrete Optimization '99
July 27, 1999

Two Recent Techniques in Approximation Algorithms

David P. Williamson
IBM Watson

Discrete Optimization '99

July 27, 1999

http://www.research.ibm.com/people/w/williamson

What...?

An α-approximation algorithm is a polynomial-time algorithm which always produces a solution of value within α times the value of an optimal solution.

α is called the performance guarantee of the algorithm: for min problems, $\alpha > 1$, and for max $\alpha < 1$.

If the algorithm is randomized, it must give a solution whose expected cost is within α of optimal.

Why...?

One possible answer to "how do we approach NP-hard optimization problems?"

There are two possible answers:

1. Use algorithms that are not provably efficient.
   - Branch-and-bound, branch-and-cut
2. Use algorithms that are not provably optimal.
   - Local search (including simulated annealing, tabu search, etc.)
   - Approximation algorithms

Approximation algorithms have been around since Vizing (1964) and Graham (1966), but in the past decade there has been a great deal of research, and two different strands have converged.

- **Complexity** Complexity theorists have developed powerful tools for showing that no α-approximation algorithms can exist unless $P = NP$.

- **Algorithms** Algorithm designers have developed techniques that apply to a wide range of problems.

Example 1: TSP

Christofides (1976) gives a $\frac{3}{2}$-approximation algorithm for the Metric TSP (i.e. edge costs obey the triangle inequality).

Theorem: (Arora, Lund, Motwani, Sudan, Szegedy 1992): There exists some $\rho > 1$ such that there is no ρ-approximation algorithm for the Metric TSP unless P=NP.

Arora (1996) and Mitchell (1996) give an approximation scheme for the Euclidean TSP (i.e. for any $\epsilon > 0$, there exists a $(1 + \epsilon)$-approximation algorithm.)

Example 2: Set cover

Johnson (1972), Chvátal (1979) give a $\ln n$-approximation algorithm for the Set Cover problem.

Theorem (Lund, Yannakakis; Feige 1996): There exists no $c \ln n$-approximation algorithm for Set Cover for $c < 1$ unless $NP \subseteq DTIME(n^{\log^k n})$.

Example 3: MAX CUT

Goemans, W '94 give a .878-approximation algorithm for MAX CUT (finding the largest cut in a graph).

Håstad '97 shows that there can be no .941-approximation algorithm for MAX CUT unless $P = NP$.

Example 4: MAX 3SAT

Johnson (1972) gives a $\frac{7}{8}$-approximation algorithm for satisfying the maximum number of clauses of a 3SAT formula (MAX E3SAT) when each clause has exactly three literals.

Theorem (Håstad 1997) There exists no $(\frac{7}{8} + \epsilon)$-approximation algorithm for MAX E3SAT unless $P = NP$.

Karloff and Zwick (1997) give a $(\frac{7}{8} - \epsilon)$-approximation algorithm for MAX 3SAT (when clauses may have 1-3 literals).

Two techniques

I. The Primal-Dual Method

II. Semidefinite Programming

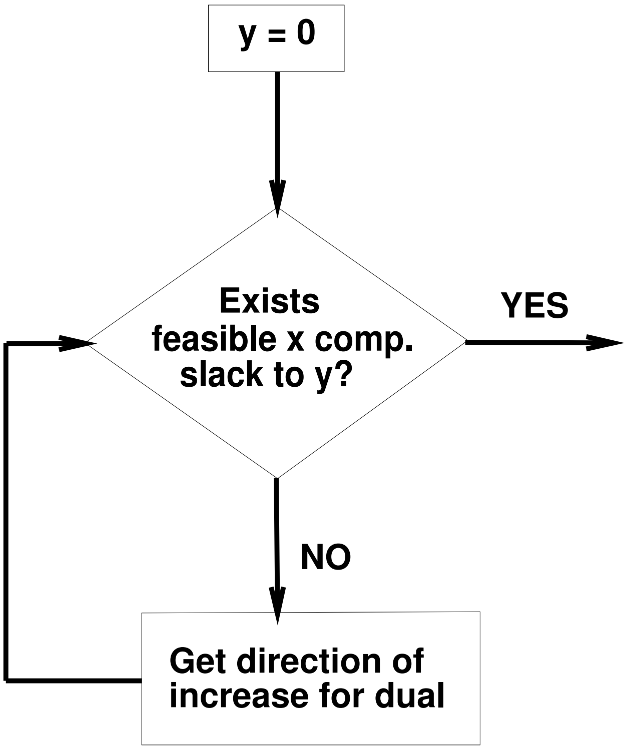
The Primal-Dual Method

$$
\begin{array}{lll}
\text{Min} & \displaystyle\sum_i c_i x_i & \\
& \displaystyle\sum_i a_{ij} x_i \geq b_j & \forall j \\
& x_i \in \{0, 1\} & \forall i
\end{array}
$$

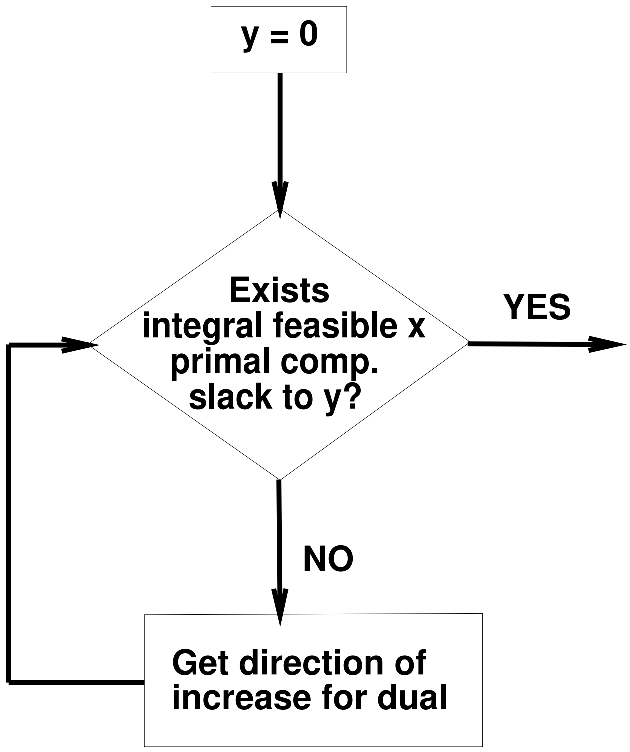
$$
\begin{array}{lll}
\text{Max} & \displaystyle\sum_j b_j y_j & \\
& \displaystyle\sum_j a_{ij} y_j \leq c_i & \forall i \\
& y_j \geq 0 & \forall j
\end{array}
$$

The primal-dual method

The standard primal-dual method for solving combinatorial optimization problems.



The Main Idea for primal-dual approximation algorithms: relax dual complementary slackness conditions (e.g. $y_j > 0 \Rightarrow \sum_i a_{ij} x_i = b_j$).



Choosing the right direction for dual increase allows for proof that the integer solution found is close to the dual solution constructed.

See Goemans & W '97, Bertsimas & Teo '95, Teo '97 for nice surveys.

$\underline{\alpha = 1}$

Min-cost spanning tree	Kruskal '56
Shortest s-t path	Dijkstra '59
Min-cost branching	Edmonds '67

$\underline{\alpha = 2}$

Vertex cover	Bar-Yehuda, Even '81
Steiner tree	various
Generalized Steiner tree	Agrawal, Klein, Ravi '95
Matching (w/ triangle inequality)	Goemans, W '95
T-joins	⋮
Non-fixed point-to-point connection	
Partition into trees/paths/cycles of at least k vertices	
Location/design problems	
Prize-collecting TSP	
⋮	
Multicuts in trees	Garg, Vazirani, Yannakakis '93
Feedback vertex sets in undirected graphs	Becker, Geiger '94 Bafna, Berman, Fujito '95

Other α

Survivable network design	W, Goemans, Mihail, Vazirani '93
	Goemans, Goldberg, Plotkin, Shmoys, Tardos, W '94
Intersecting contra-polymatroids	Bertsimas, Teo '95
Feedback vertex sets in planar graphs	Goemans, W '96
Steiner trees in special graphs	Rajagopalan, Vazirani '98
Uncapacitated facility location	Jain, Vazirani '99
k-median	Jain, Vazirani '99
	Guha, Charikar '99

Experimental studies

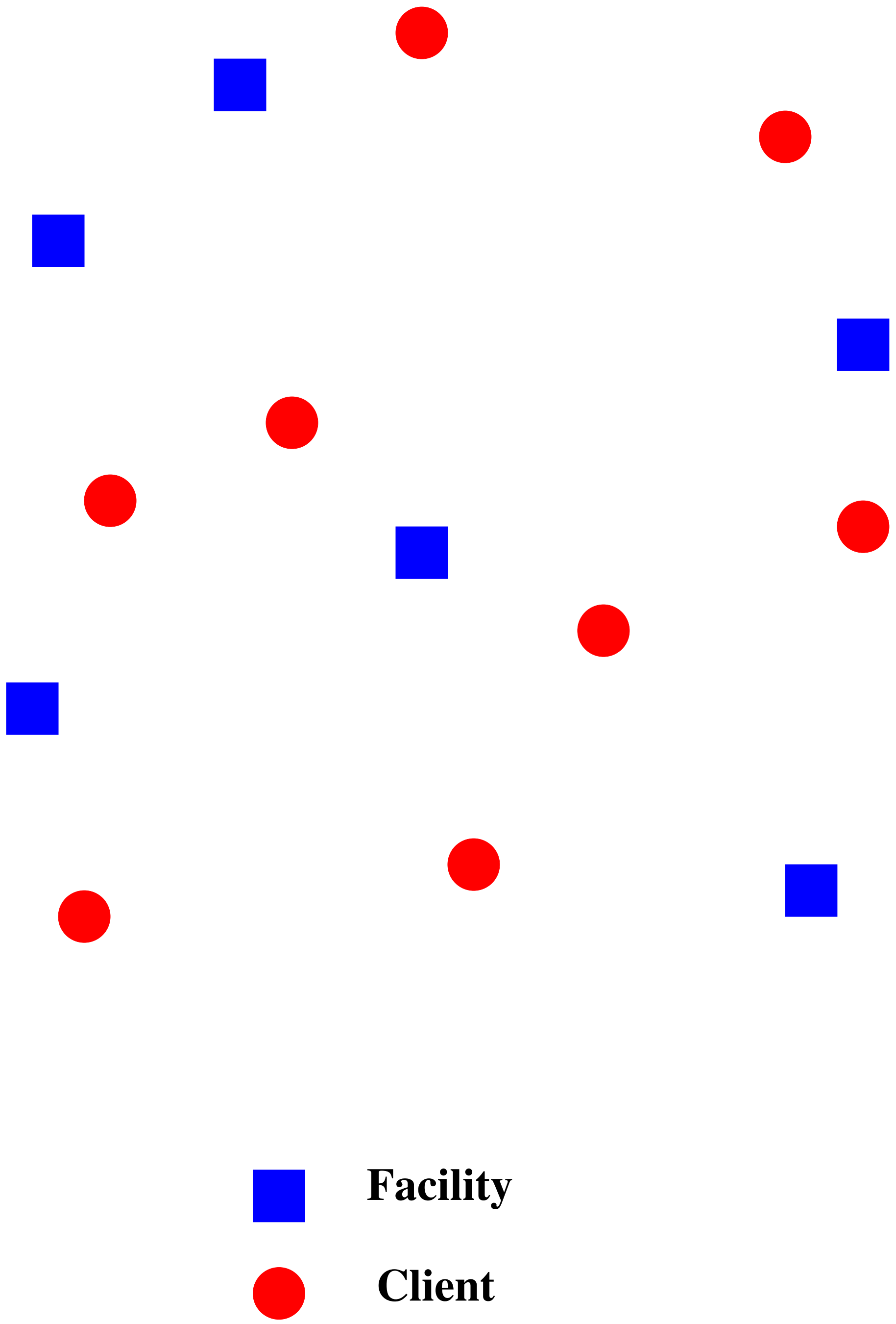
Euclidean matching	Goemans, W '94
Steiner trees	Hall '95
Generalized Steiner trees	Hu & Wein
Network design problems	Mihail, Mostrel, Dean, Shallcross '96
Feedback vertex set	Becker & Geiger '96
Prize-collecting Steiner tree	Johnson, Minkoff, Phillips '99

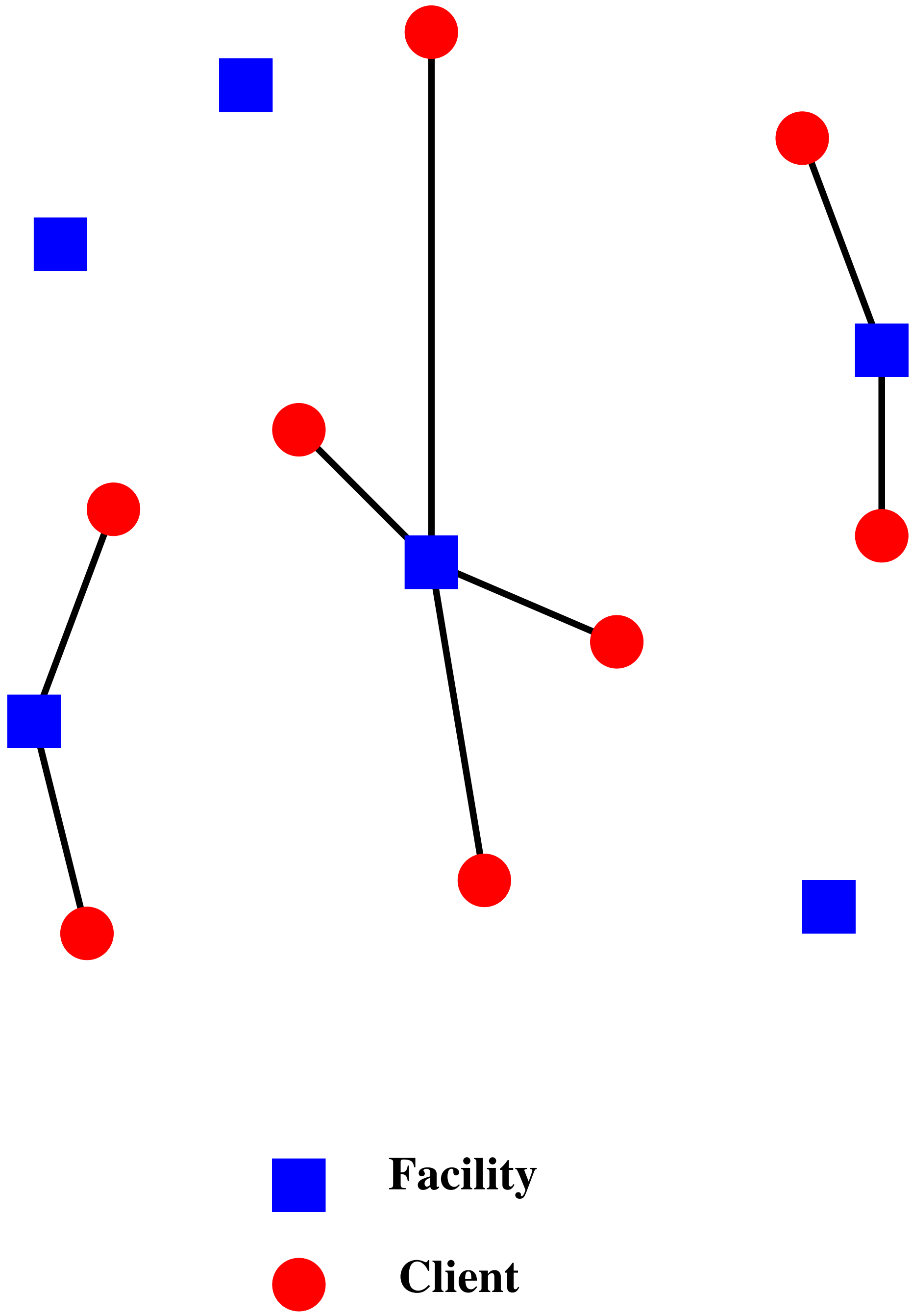
Example: Uncapacitated Facility Location Problem

- **Input:**
  - A set of locations V.
  - Facilities $F \subset V$, and cost f_i for $i \in F$.
  - Clients $D = V - F$, and cost c_{ij} for assigning client $j \in D$ to facility $i \in F$.
  - Edge costs obey the triangle inequality.

- **Output:** A set of facilities F' and an assignment of clients to facilities in F' that minimizes total cost (Facility + Assignment).

- NP-hard (Cornuéjols, Nemhauser, and Wolsey '90).


Facility
Client


Facility
Client

An integer programming formulation

A formulation of the problem (Balinski 1966):

$$
\begin{array}{lll}
\text{Min} & \displaystyle\sum_{i \in F} f_i y_i + \sum_{i \in F, j \in D} c_{ij} x_{ij} & \\
& \displaystyle\sum_i x_{ij} = 1 & \forall j \in D, \\
& x_{ij} \leq y_i & \forall i \in F, j \in D, \\
& y_i \in \{0,1\} & \forall i \in F, \\
& x_{ij} \in \{0,1\} & \forall i \in F, j \in D,
\end{array}
$$

where

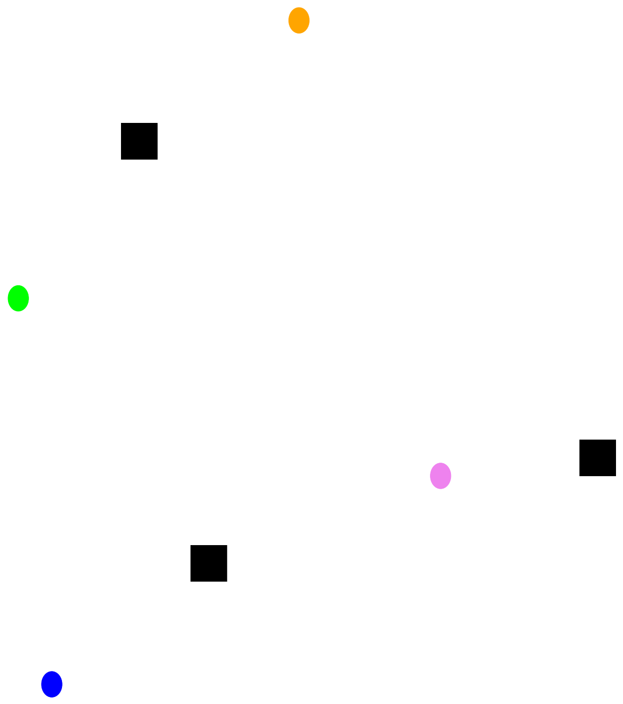
- $y_i = 1$ iff facility i is opened;
- $x_{ij} = 1$ iff client j is assigned to facility i.

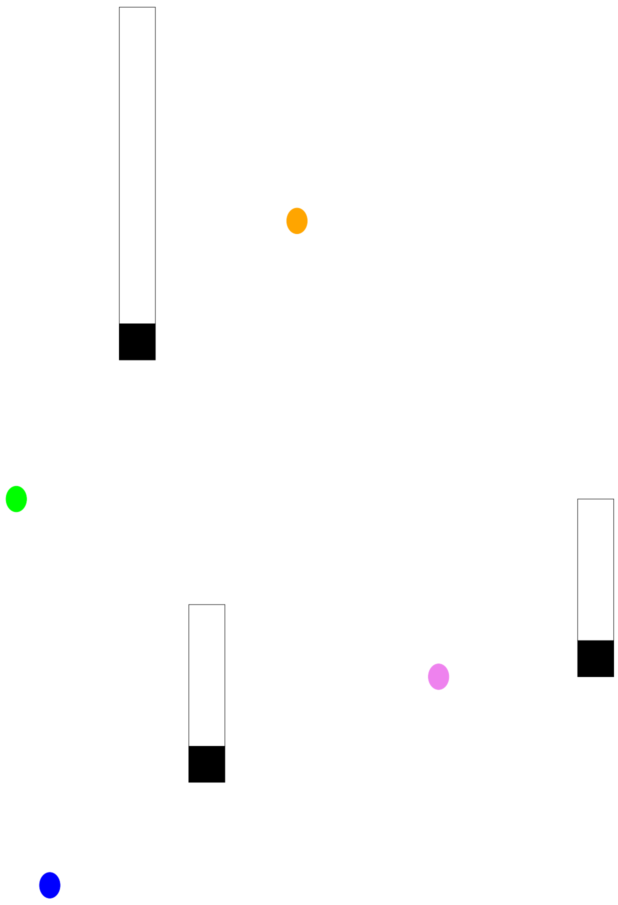
Dual and Algorithm

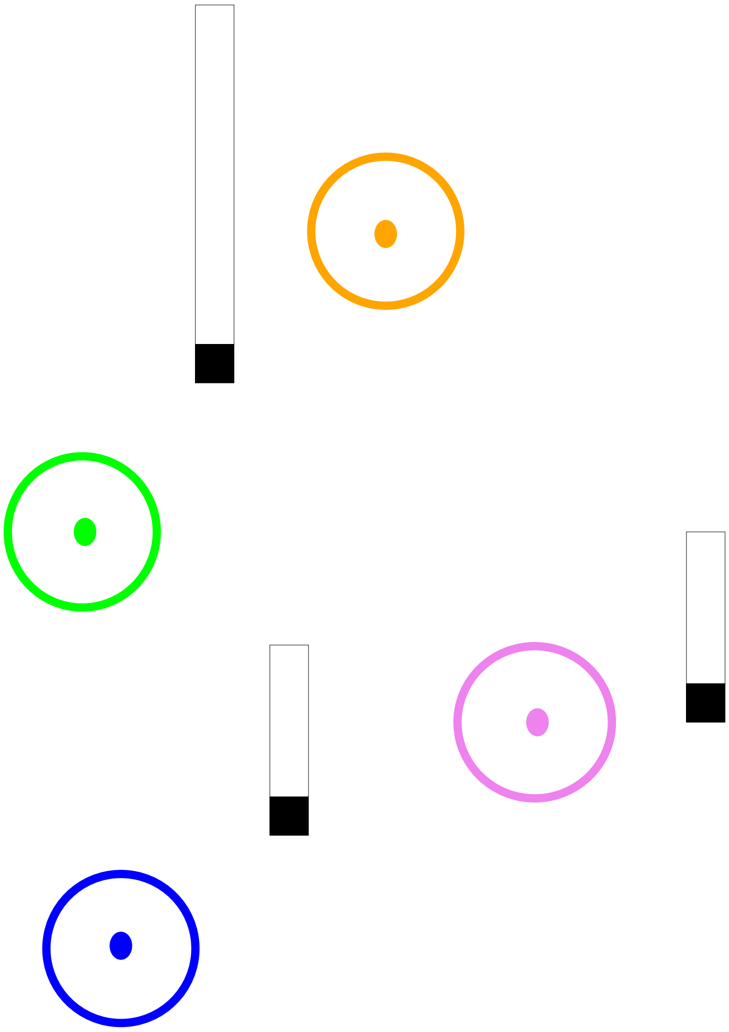
$$
(D) \qquad
\begin{array}{lll}
\text{Max} & \displaystyle\sum_{j \in D} v_j & \\
& \displaystyle\sum_{j \in D} w_{ij} \leq f_i & \forall i \in F, \\
& v_j - w_{ij} \leq c_{ij} & \forall i \in F, j \in D, \\
& w_{ij} \geq 0 & \forall i \in F, j \in D.
\end{array}
$$

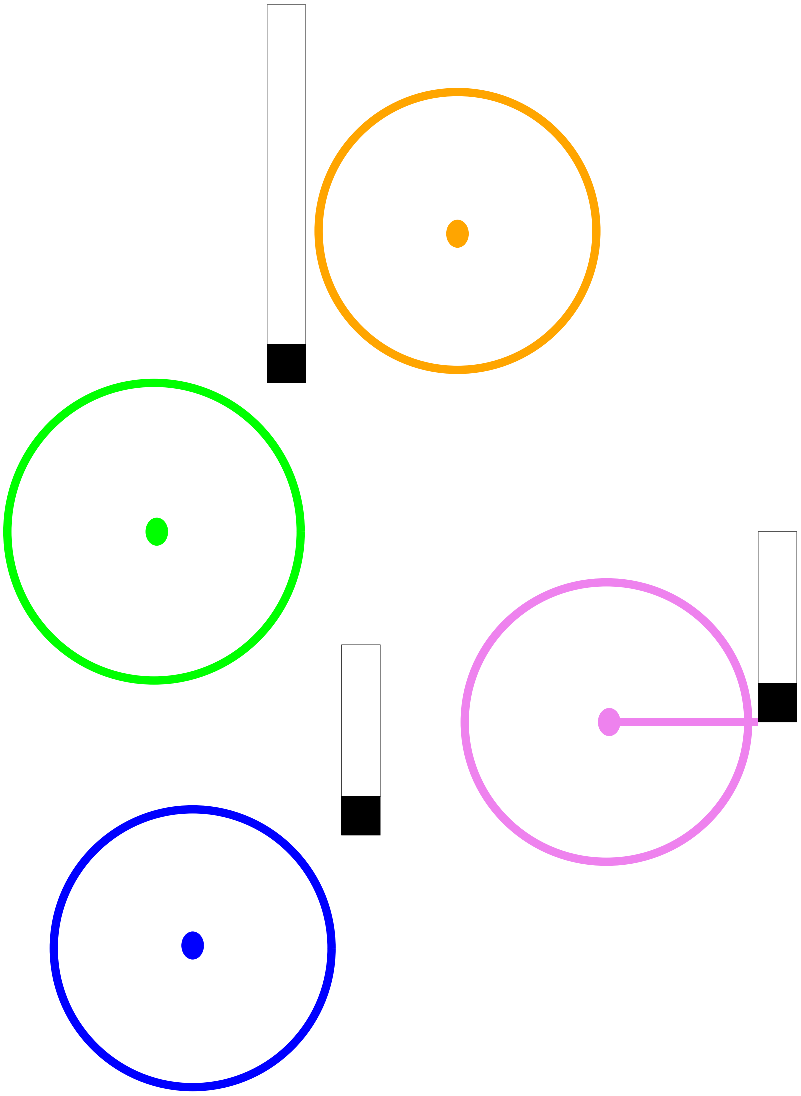
1. Set $v \leftarrow 0$, $w \leftarrow 0$
2. Increase v_j variables uniformly and w_{ij} as necessary, until cannot increase without violating constraints.
3. Open a subset of the facilities i such that $\sum_{j \in D} w_{ij} = f_i$; assign clients to facilities.

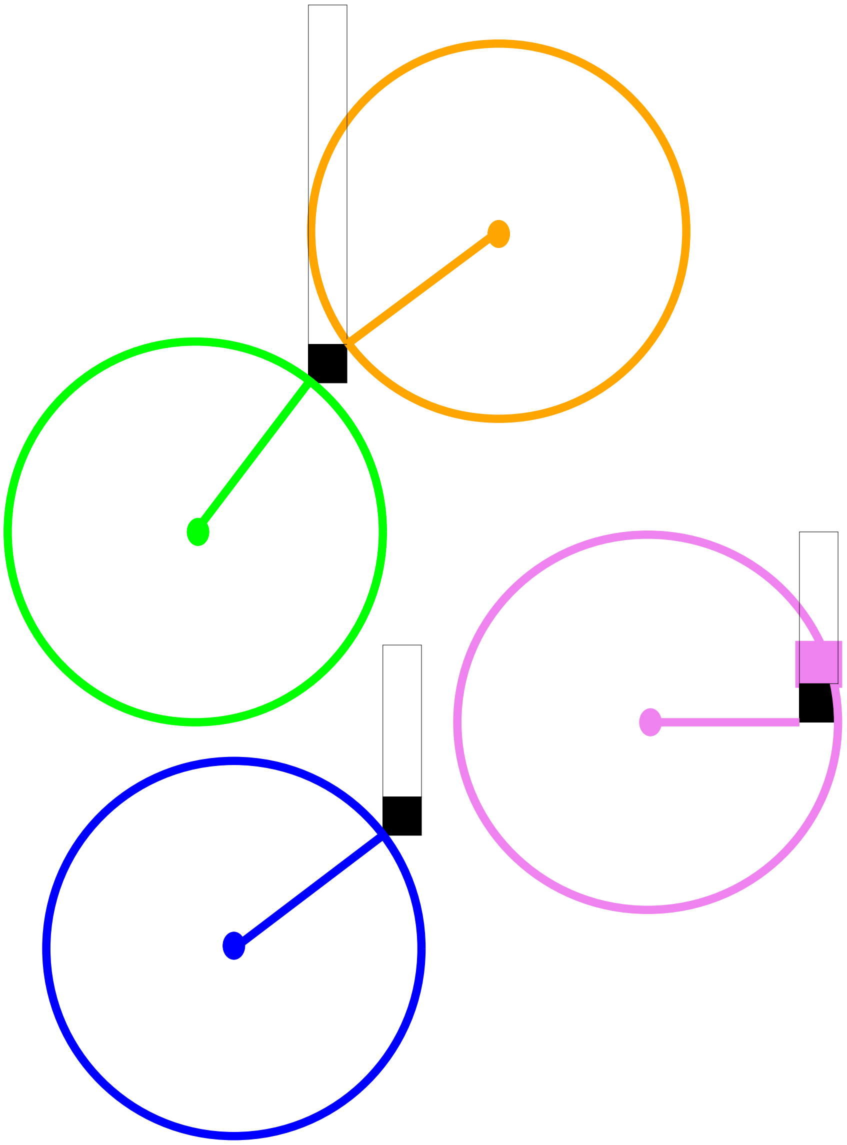
(Jain, Vazirani '99; compare to Erlenkotter '78)

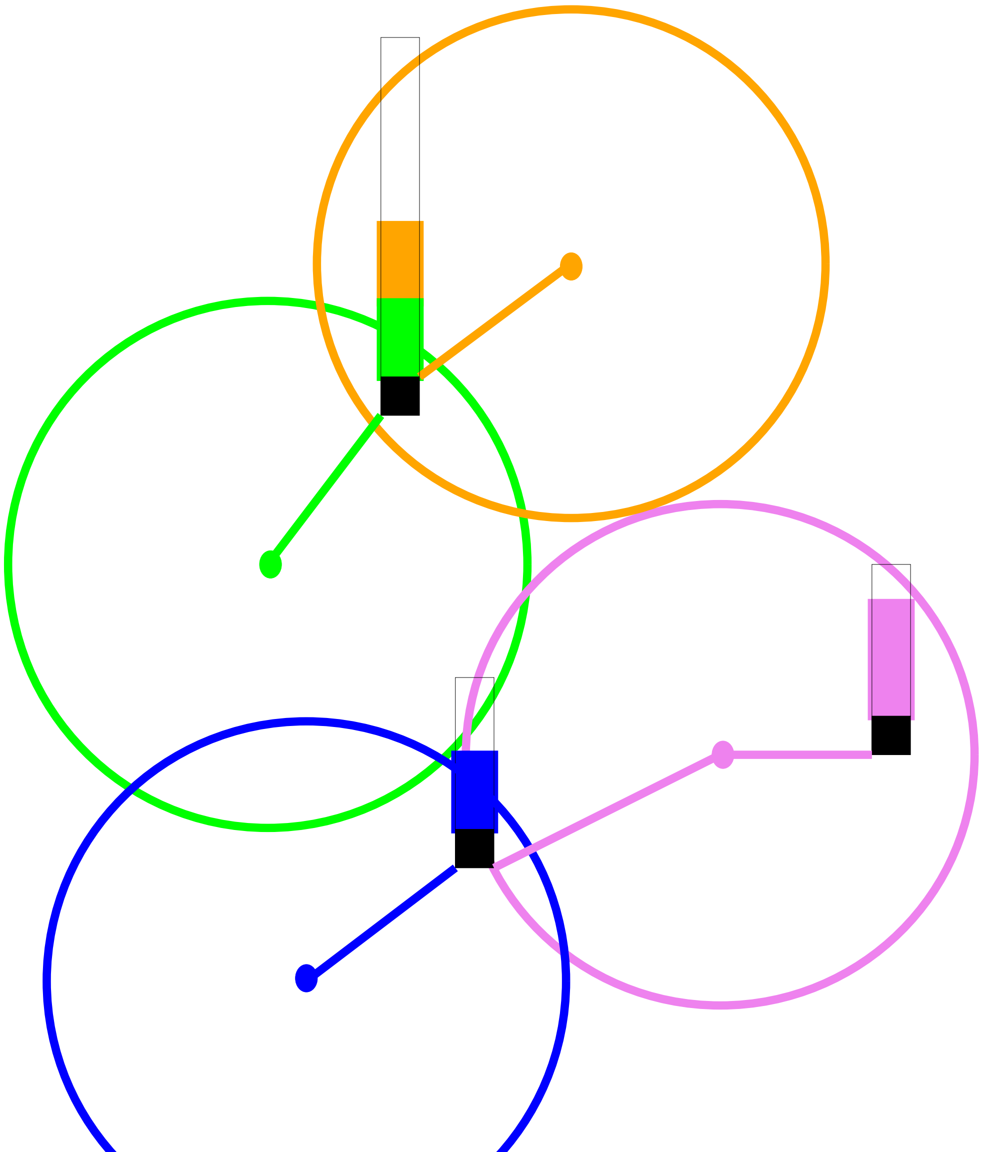


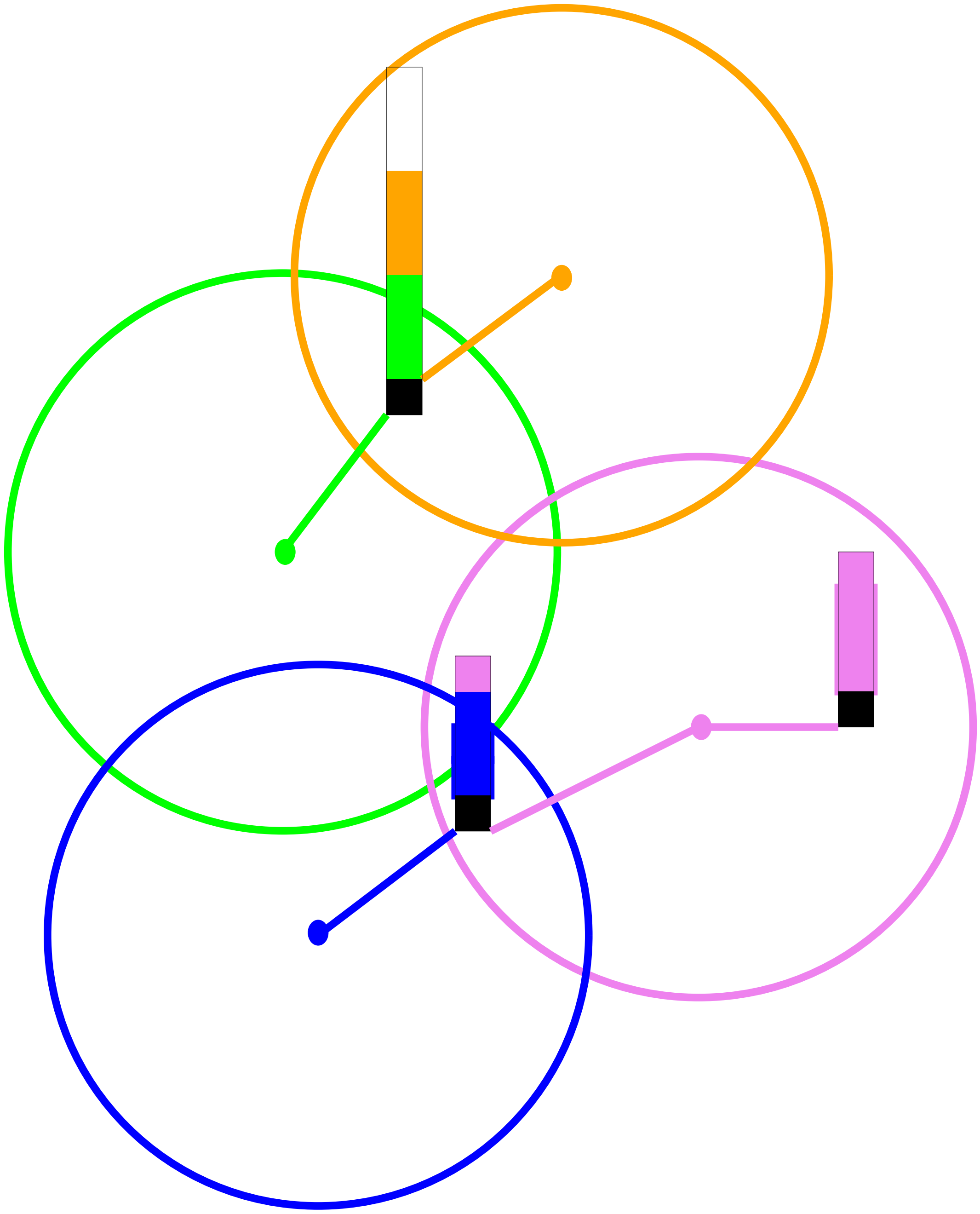


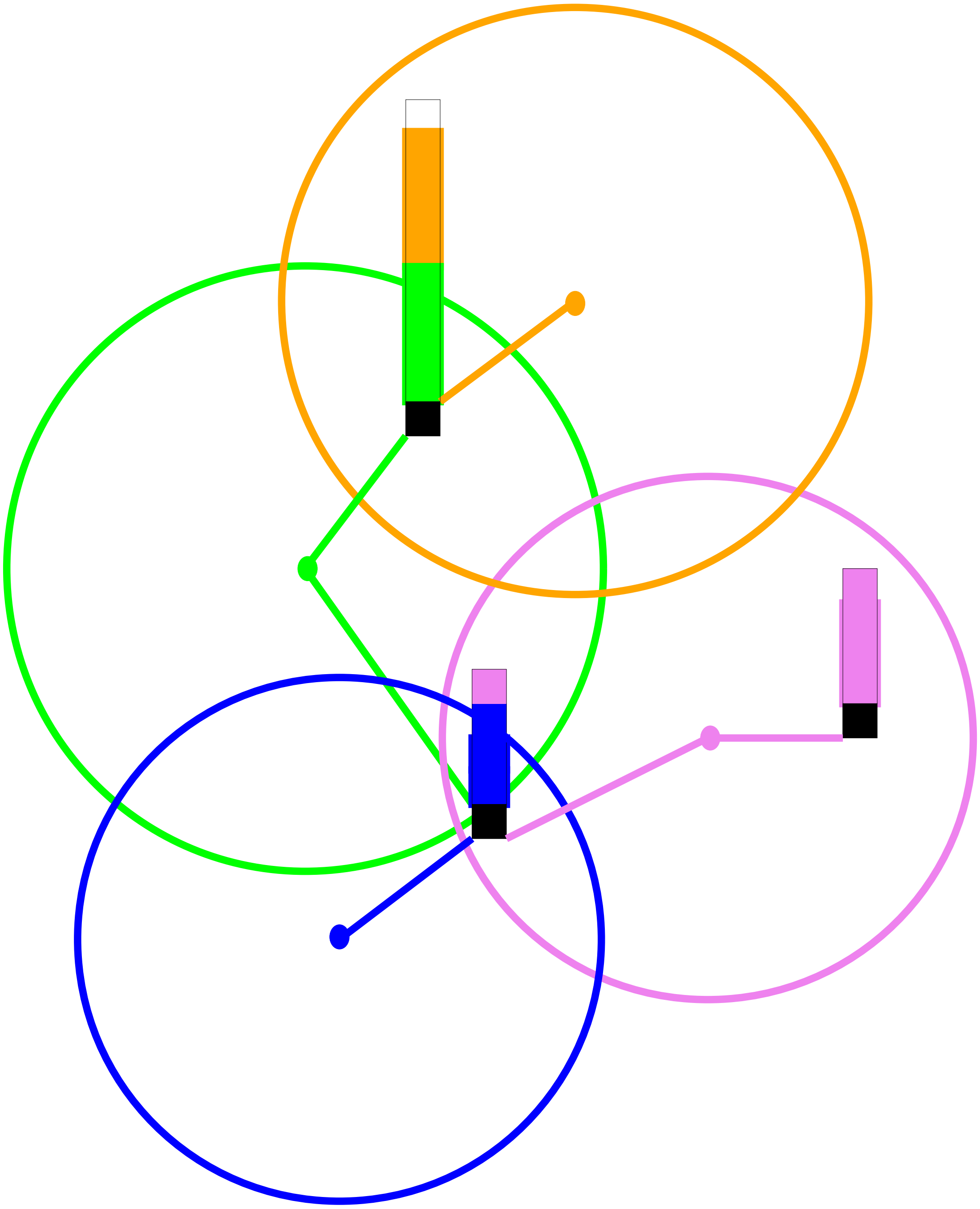


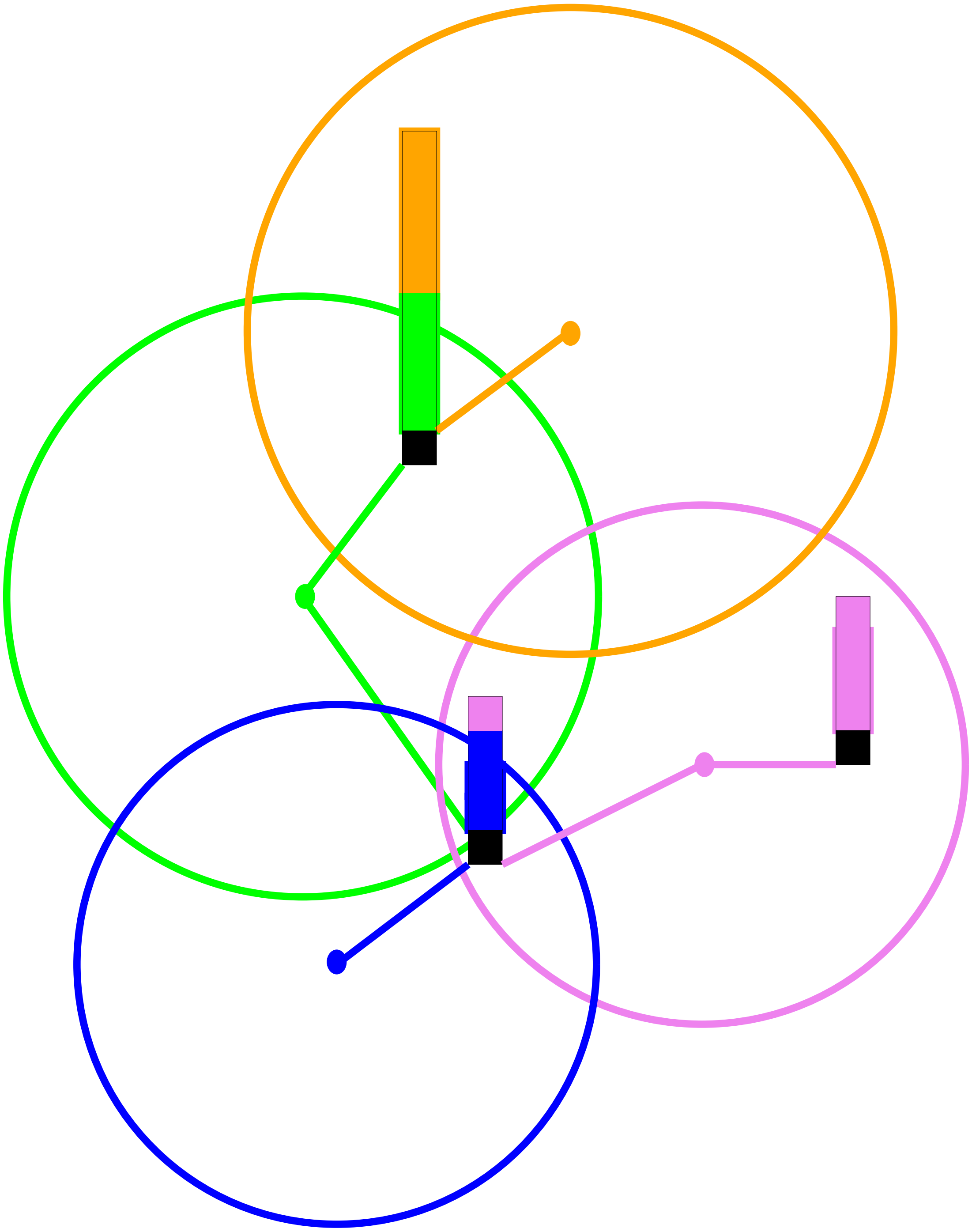












3. Consider facilities i with $\sum_{j \in D} w_{ij} = f_i$ in order inequalities became tight. Open i if no neighboring facility of a neighboring client is open.

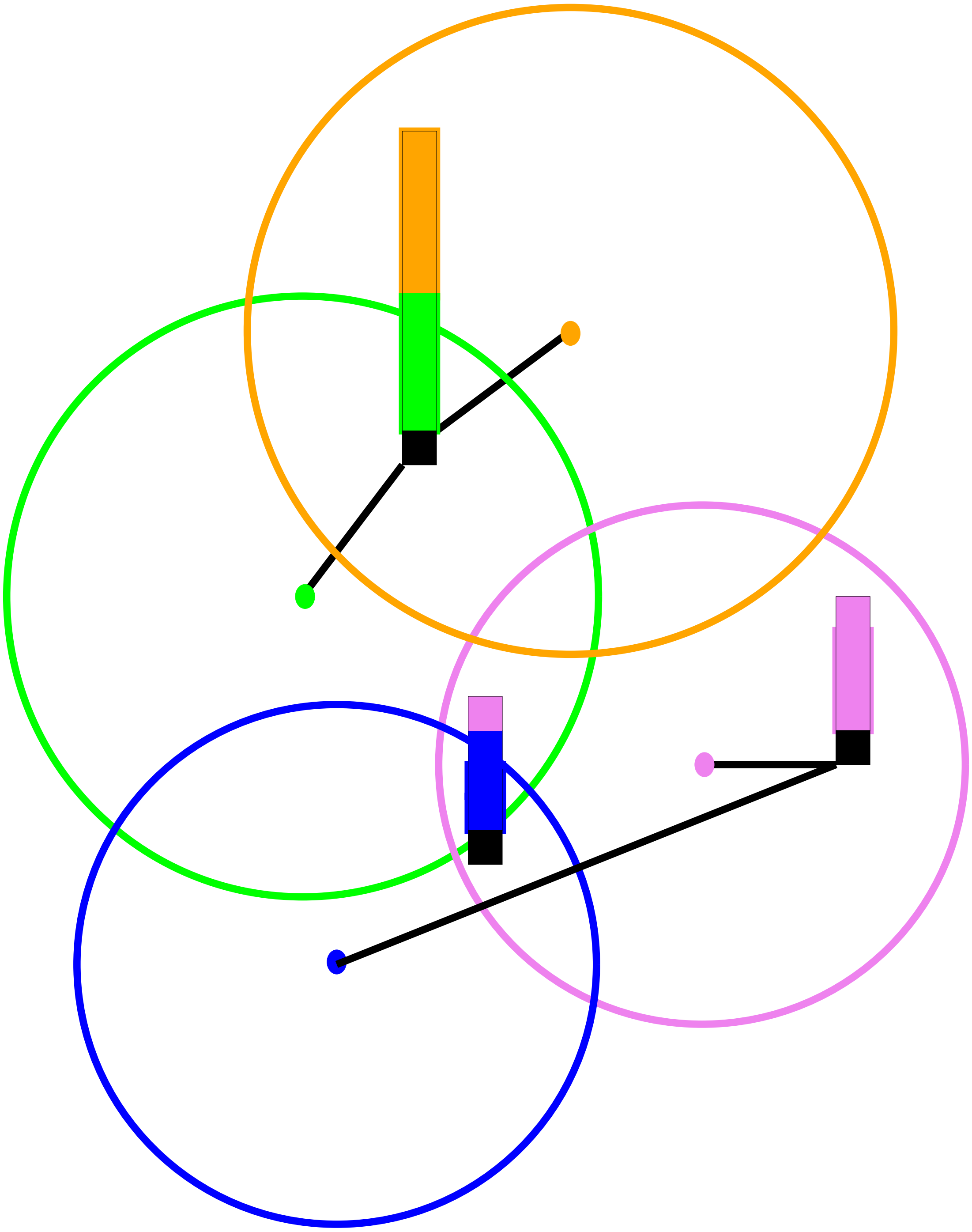
   Assign client j to any open neighbor, or to the first opened neighbor of a neighbor of a neighbor.

Theorem (Jain, Vazirani '99)
For this integer solution $(\bar{x}, \bar{y})$,

$$\begin{aligned} \sum_{i \in F, j \in D} c_{ij}\bar{x}_{ij} + \sum_i f_i \bar{y}_i &\leq 3 \sum_j v_j \\ &\leq 3 \cdot OPT. \end{aligned}$$

Using different technique, get 1.728-approx. alg. (Chudak, Shmoys '98; Guha, Charikar '99)

More complicated applications possible?



Semidefinite programming

A semidefinite program has the form

$$\begin{array}{lll} \text{Max/Min} & C \bullet X & \\ & A_k \bullet X = b_k & \forall k \\ & X \succeq 0 & X \text{ symmetric.} \end{array}$$

where $A \bullet B = \sum_{i,j} a_{ij} b_{ij}$ for $A, B \in \Re^{n \times n}$, and $A \succeq B$ iff $A - B$ is positive semidefinite.

Definition: A matrix $X \in \Re^{n \times n}$ is positive semidefinite iff $a^T X a \geq 0$ for all $a \in \Re^n$.

Useful facts

The following three statements are equivalent for a symmetric matrix X:

1. X is positive semidefinite;
2. X has only non-negative eigenvalues;
3. $X = V^T V$ for some matrix $V \in \Re^{m \times n}$.

Semidefinite programming (SDP) is polynomial-time solvable to within an additive error of ϵ using either:

1. The ellipsoid method: if v is eigenvector of negative eigenvalue, then $v^T X v \geq 0$ is a separating hyperplane. (Grötschel, Lovász, Schrijver '88).
2. Interior-point methods (Nesterov & Nemirovskii; Alizadeh '95)

Using these facts, semidefinite programming is equivalent to a kind of "vector programming".

$$
\begin{array}{lll}
\text{Max/Min} & C \bullet X & \\
 & A_k \bullet X = b_k & \forall k \\
 & X \succeq 0 & X \text{ symm}
\end{array}
$$

is equivalent to

$$
\begin{array}{lll}
\text{Max/Min} & \sum_{i,j} c_{ij}(v_i \cdot v_j) & \\
 & \sum_{i,j} a_{ijk}(v_i \cdot v_j) = b_k & \forall k \\
 & v_i \in \Re^n &
\end{array}
$$

since $X = V^T V$, v_i the ith column of V iff X is symmetric, psd.

The technique

Model problem of interest as a binary quadratic program:

$$\text{Max} \quad \sum_{i,j} a_{ij} x_i x_j$$
$$x \in \{-1, 1\}^n.$$

Relax to a vector program:

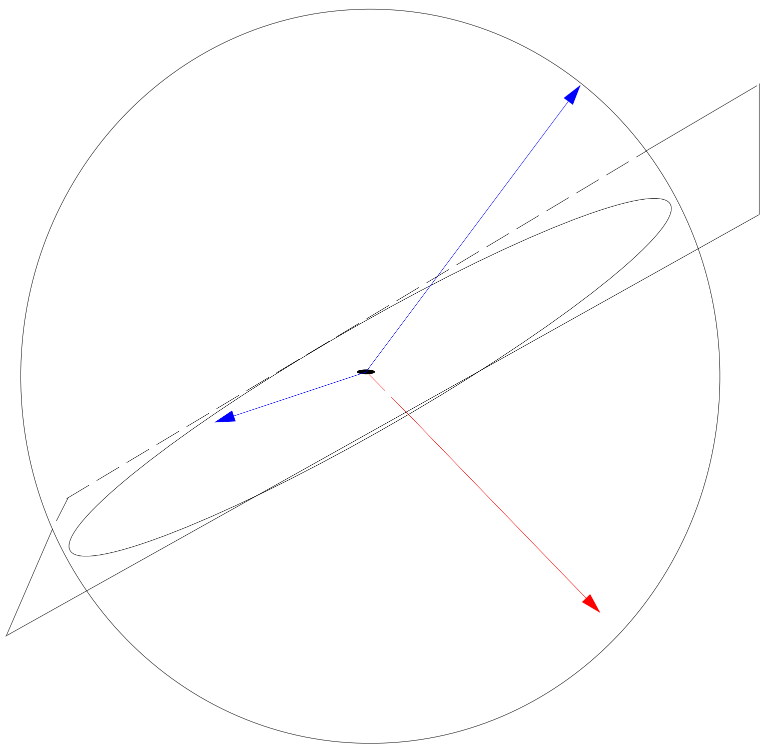
$$\text{Max} \quad \sum_{i,j} a_{ij} (v_i \cdot v_j)$$
$$v_i \cdot v_i = 1$$
$$v_i \in \Re^n$$

Choose random vector r uniformly from unit sphere S^{n-1}. Get solution

$$\bar{x}_i = \begin{cases} 1 & \text{if } v_i \cdot r \geq 0 \\ -1 & \text{if } v_i \cdot r < 0 \end{cases}$$

Geometric interpretation

Vector program is a relaxation of the case where x_i are one-dimensional unit vectors to n dimensions. Choosing random r is the same as choosing a random hyperplane.



Applications

Maximum cut	Goemans, W '94
Maximum directed cut	Goemans, W '94
	Feige, Goemans '94
MAX 2SAT	Goemans, W '94
	Feige, Goemans '94
MAX SAT	Goemans, W '94
	Asano, Ono, Hirata '96
	Asano '97
	Zwick '99
	Asano, W '99
Maximum k-cut	Frieze, Jerrum '94
Maximum bisection	Frieze, Jerrum '94
	Ye '99
Minimum k-coloring	Karger, Motwani, Sudan '95
Betweeness	Chor, Sudan '95
Quadratic programming	Nesterov '97
	Ye '97,'98
	Zhang '98

Applications

MAX 3SAT	Trevisan, Sorkin, Sudan, W '96
	Karloff, Zwick '97
MAX 3CSP	Zwick '98
Scheduling	Skutella '98
Linear Eqs mod p	Andersson, Engebretsen, Håstad '98
Minimum Bandwidth	Blum, Konjevod, Ravi, Vempala '98
MAX 4SAT	Halperin, Zwick '99

<u>Experimental Studies</u>

Maximum cut	Goemans, W '95
	Homer, Peinado '95
	Berry, Goldberg '95
	Benson, Ye, Zhang '97
	Burer, Monteiro '98

Quadratic programming

Nesterov '97 gives a $\frac{2}{\pi}$-approximation algorithm to the following quadratic programming problem with $A = (a_{ij})$ is symmetric, psd.

$$\begin{aligned} \text{Max} \quad & \sum_{i,j} a_{ij} x_i x_j \\ & x_i \in \{-1, 1\} \qquad\qquad \forall i \end{aligned}$$

Note that

$$\sum_{i,j} a_{ij} x_i x_j = x^T A x \geq 0.$$

Relaxation

Consider the vector programming relaxation

$$
\begin{array}{lll}
\text{Max} & \sum_{i,j} a_{ij}(v_i \cdot v_j) & \\
 & v_i \cdot v_i = 1 & \forall i, \\
 & v_i \in \Re^n, &
\end{array}
$$

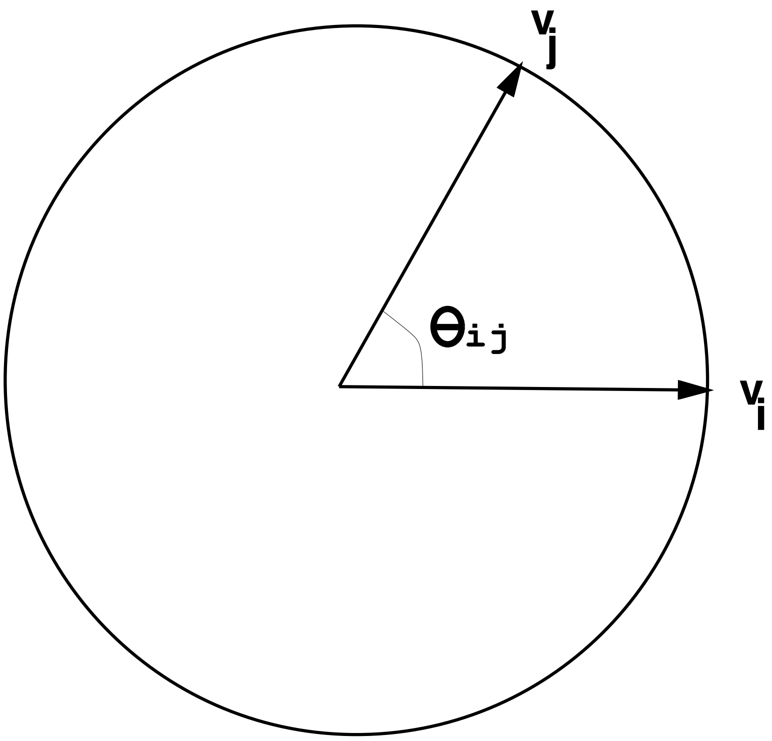
or, equivalently,

$$
\begin{array}{lll}
\text{Max} & A \bullet X & \\
 & x_{ii} = 1 & \forall i, \\
 & X \succeq 0. &
\end{array}
$$

Let X^* and v_i^* denote the optimal solution to the relaxation, and obtain the solution $\bar{x}$ as indicated.

Lemma 1

$$\Pr[\bar{x}_i \bar{x}_j = -1] = \frac{1}{\pi}\theta_{ij} = \frac{1}{\pi}\arccos(v_i^* \cdot v_j^*) = \frac{1}{\pi}\arccos(x_{ij}^*)$$



Analysis

Then

$$
\begin{aligned}
E[\bar{x}_i \bar{x}_j] &= \left[\left(1 - \frac{1}{\pi}\arccos(x^*_{ij})\right) - \frac{1}{\pi}\arccos(x^*_{ij})\right] \\
&= \left[1 - \frac{2}{\pi}\arccos(x^*_{ij})\right] \\
&= \left[1 - \frac{2}{\pi}\left(\frac{\pi}{2} - \arcsin(x^*_{ij})\right)\right] \\
&= \frac{2}{\pi}\arcsin(x^*_{ij}),
\end{aligned}
$$

using the fact that $\arcsin(x) + \arccos(x) = \frac{\pi}{2}$.

Analysis cont.

Let $\arcsin[X] = (\arcsin(x_{ij}))$ and $\bar{X} = (\bar{x}_i \bar{x}_j)$.

The expected value of our solution is

$$E[\sum_{ij} a_{ij} \bar{x}_i \bar{x}_j] = \frac{2}{\pi} \sum_{ij} a_{ij} \arcsin(x^*_{ij}),$$

or

$$E[A \bullet \bar{X}] = \frac{2}{\pi} A \bullet \arcsin[X^*].$$

The value of the SDP relaxation is

$$\sum_{ij} a_{ij} x^*_{ij} = A \bullet X^* \geq OPT.$$

If

$$E[A \bullet \bar{X}] - \frac{2}{\pi} A \bullet X^* \geq 0,$$

then have a $\frac{2}{\pi}$-approximation algorithm.

Analysis cont.

Lemma $\arcsin[X] - X \succeq 0$ if $|x_{ij}| \leq 1$ for all i, j.

Fact If $C \succeq 0$, $D \succeq 0$, then $C \bullet D \geq 0$.

$$\begin{aligned}
E[A \bullet \bar{X}] - \frac{2}{\pi} A \bullet X^* &= \frac{2}{\pi}(A \bullet \arcsin[X^*] - A \bullet X^*) \\
&= \frac{2}{\pi} A \bullet (\arcsin[X^*] - X^*) \\
&\geq 0,
\end{aligned}$$

it follows that

$$E[A \bullet \bar{X}] \geq \frac{2}{\pi} A \bullet X^* \geq \frac{2}{\pi} OPT.$$

Other techniques: LP rounding

Idea: Solve LP relaxation of IP formulation of the problem. Round all decision vars above a certain threshold to the nearest integer. Repeat on remaining subproblem as necessary.
(Hochbaum '82)

Application: Survivable network design.
Jain '98 shows that at least one variable x_e in the LP relaxation has $x_e \geq 1/2$, implies a 2-approximation algorithm.

Other techniques: Randomized rounding

Idea: Solve LP relaxation of IP formulation of the problem. Create integer solution $\bar{x}$ from LP soln. x^* by setting $\bar{x}_i = 1$ with probability x_i^*.
(Raghavan, Thompson '87)

Application: Uncapacitated facility location.
Chudak and Shmoys '98 show that interleaving randomized rounding on facility vars. with a decomposition technique gives a performance guarantee of $1 + \frac{2}{e} \approx 1.736$.

“New” technique: Local search?

Idea: Apply local search techniques.

Application: Uncapacitated/capacitated facility location. Add/drop/swap facilities as it improves the cost of the solution.

- Guha, Charikar '99 show that this gives performance guarantee of 3 for uncapacitated facility location.

- Korupolu, Plaxton, Rajaraman '98 show that this gives performance guarantee of 8 for capacitated facility location (improved to 6 in Chudak, W '99).

Question: What's the lower bound?