Hospital layout optimization using GRASP with path-relinking

International conference on system analysis tools for better health care delivery: A new engineering/health care partnership ♣ Gainesville, Florida ♣ March 24-26. 2010

Mauricio G. C. Resende
AT&T Labs Research
Florham Park, New Jersey
mgcr@att.com

Joint work with
Ricardo M. A. Silva (UFLA, Brazil & post-doc at AT&T Labs Research)

Summary

- Modeling hospital layout via quadratic assignment
- Modeling hospital layout via generalized quadratic assignment
- Generalized quadratic assignment problem (GQAP)
- GRASP with path-relinking for GQAP
  - GRASP construction
  - Local search
  - Path-relinking
- Experimental results

Hospital layout as a QAP [Elshafei, 1977]

- Assign N facilities (surgery, ICU, recovery, ...) to N locations in the hospital
  - Each facility is assigned to a unique location
  - Each location has only one facility assigned to it
- Given:
  - Number of patients that move between each pair (i,j) of facilities (in some time period): P(i,j)
  - Distance between each pair of locations: D(k,l)
- Minimize average distance traveled by patients

Hospital layout as a QAP [Elshafei, 1977]

- assignment array π:
  - [$\pi(i) = j$ ⇔ facility i is assigned to location j]
- $P[i,j] \times D[\pi(i), \pi(j)]$
  - Total distance traveled by patients between facilities i and j that are assigned to locations $\pi(i)$ and $\pi(j)$, respectively

Hospital layout as a QAP [Elshafei, 1977]

Find the assignment vector π from all possible permutations Π_N of {1, 2, ..., N} that minimizes:

$$\Sigma_{i,j} P[i,j] \times D[\pi(i), \pi(j)]$$

Hospital layout as a QAP [Elshafei, 1977]

Find the assignment vector π from all possible permutations Π_N of {1, 2, ..., N} that minimizes:

$$\Sigma_{i,j} P[i,j] \times D[\pi(i), \pi(j)]$$

QAP's are one of the most computationally difficult classes of combinatorial optimization problems: Instances of size N=20 are considered challenging for exact methods.

Hospital layout as a QAP [Elshafei, 1977]

Find the assignment vector π from all possible permutations Π_N of {1, 2, ..., N} that minimizes:

$$\sum_{i,j} P[i,j] \times D[\pi(i), \pi(j)]$$

Heuristics are optimization methods that find good, though not provably optimal solutions to combinatorial optimization problems like the QAP.

Hospital layout as a QAP [Elshafei, 1977]

Find an assignment vector π from all possible permutations Π_N of {1, 2, ..., N} that minimizes:

$$\Sigma_{i,j} P[i,j] \times D[\pi(i), \pi(j)]$$

Since the 1990s, many effective heuristics have been developed for the QAP. Examples: simulated annealing, tabu search, genetic algorithms, ant colony optimization, and GRASP.

Hospital layout as a QAP [Elshafei, 1977]

The main drawback of the QAP model is that it assumes that it does not take into account that facilities have different dimensions and that they must be assigned to locations that can accommodate them.

Hospital layout as a QAP [Elshafei, 1977]

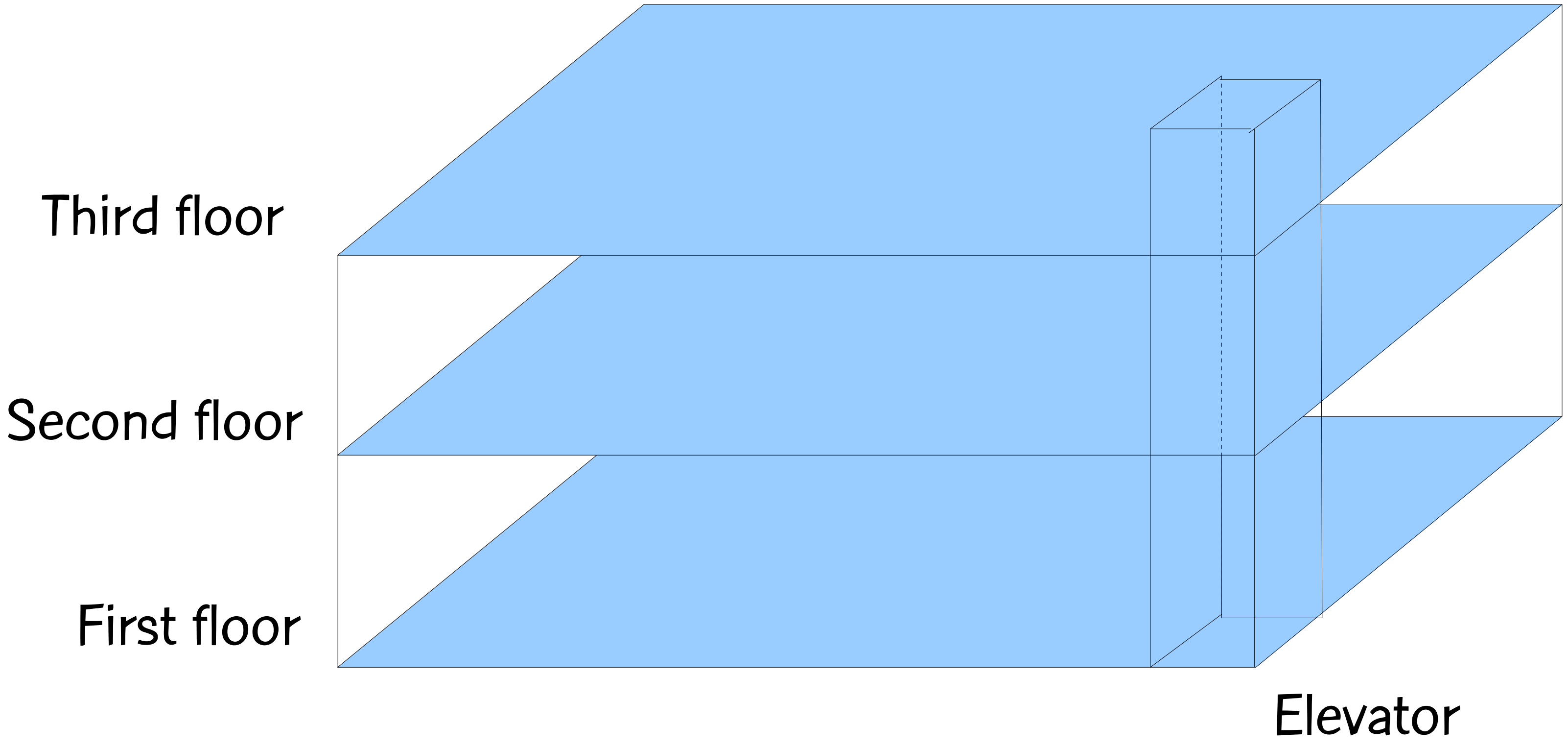
The main drawback of the QAP model is that it assumes that it does not take into account that facilities have different dimensions and that they must be assigned to locations that can accommodate them.

The generalized QAP model addresses this.

Hospital layout as a GQAP

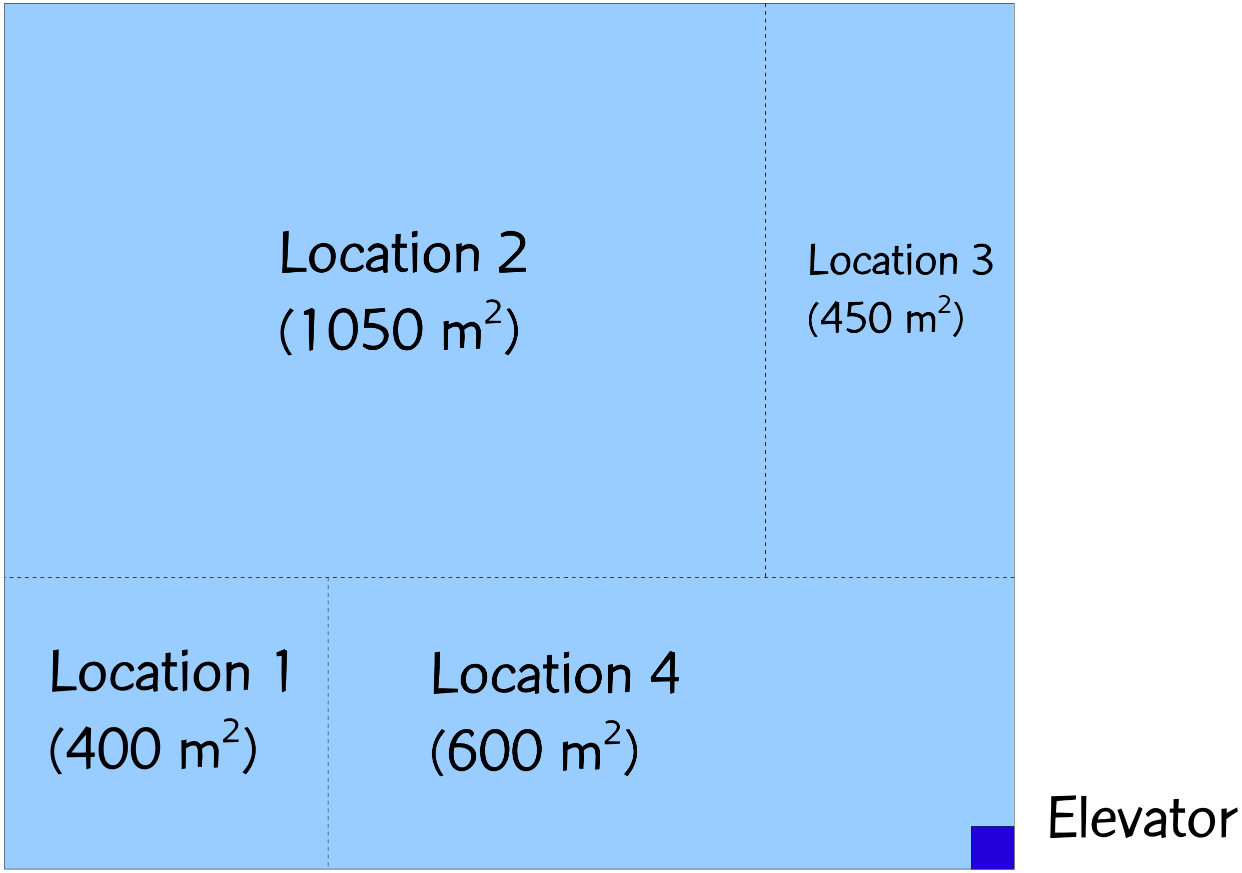
- The GQAP is similar to the QAP except that
  - Facilities have an associated area
  - Locations have an associated total available area
- Assign facilities to locations minimizing the average distance traveled by patients such that
  - Sum of areas of facilities assigned to a location does not exceed the total available area of the location
  - More than one facility can be assigned to a location.
  - No facility can be be assigned to a location.

Hospital layout as a GQAP



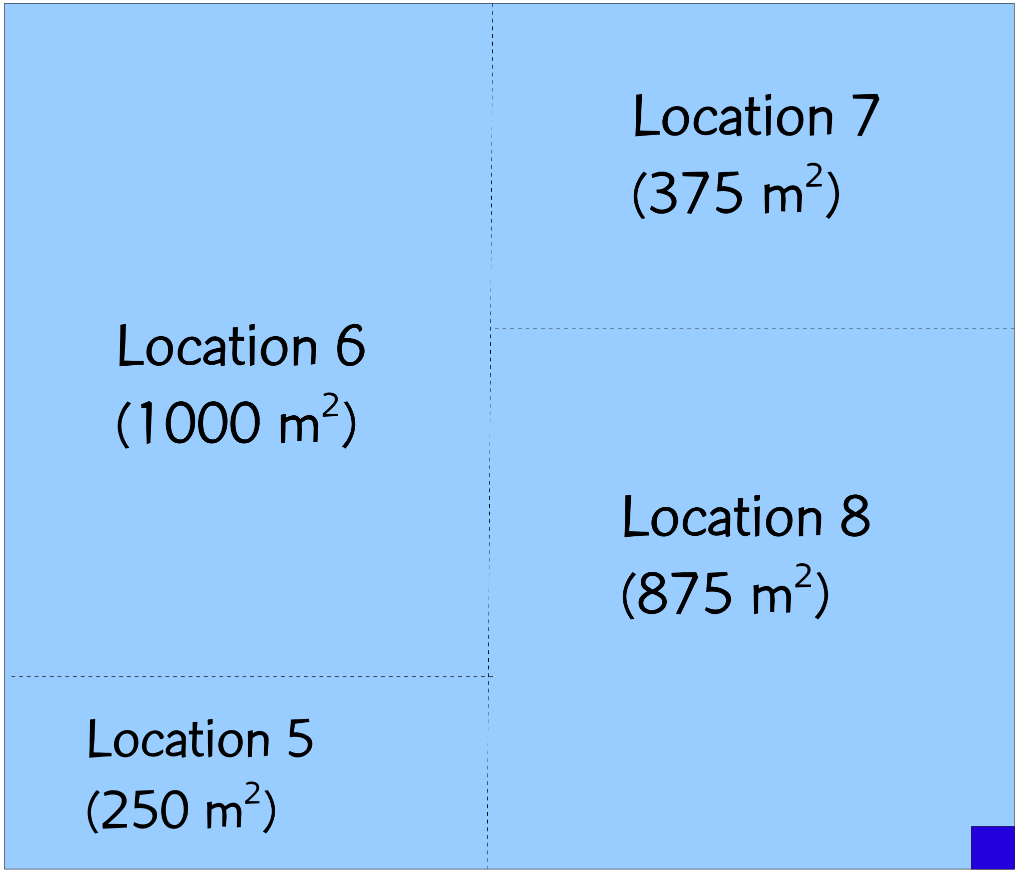
Hospital layout as a GQAP

First floor



Hospital layout as a GQAP

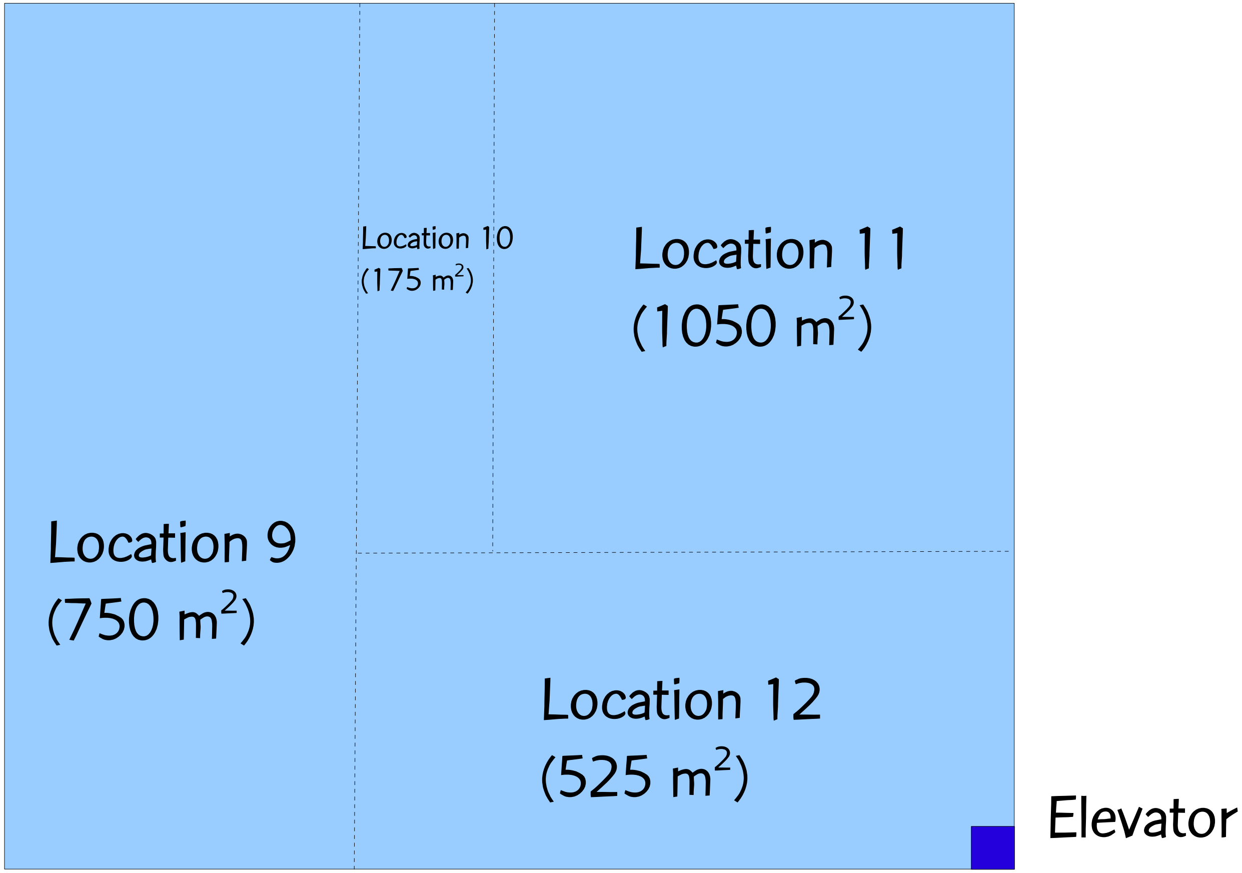
Second floor



Elevator

Hospital layout as a GQAP

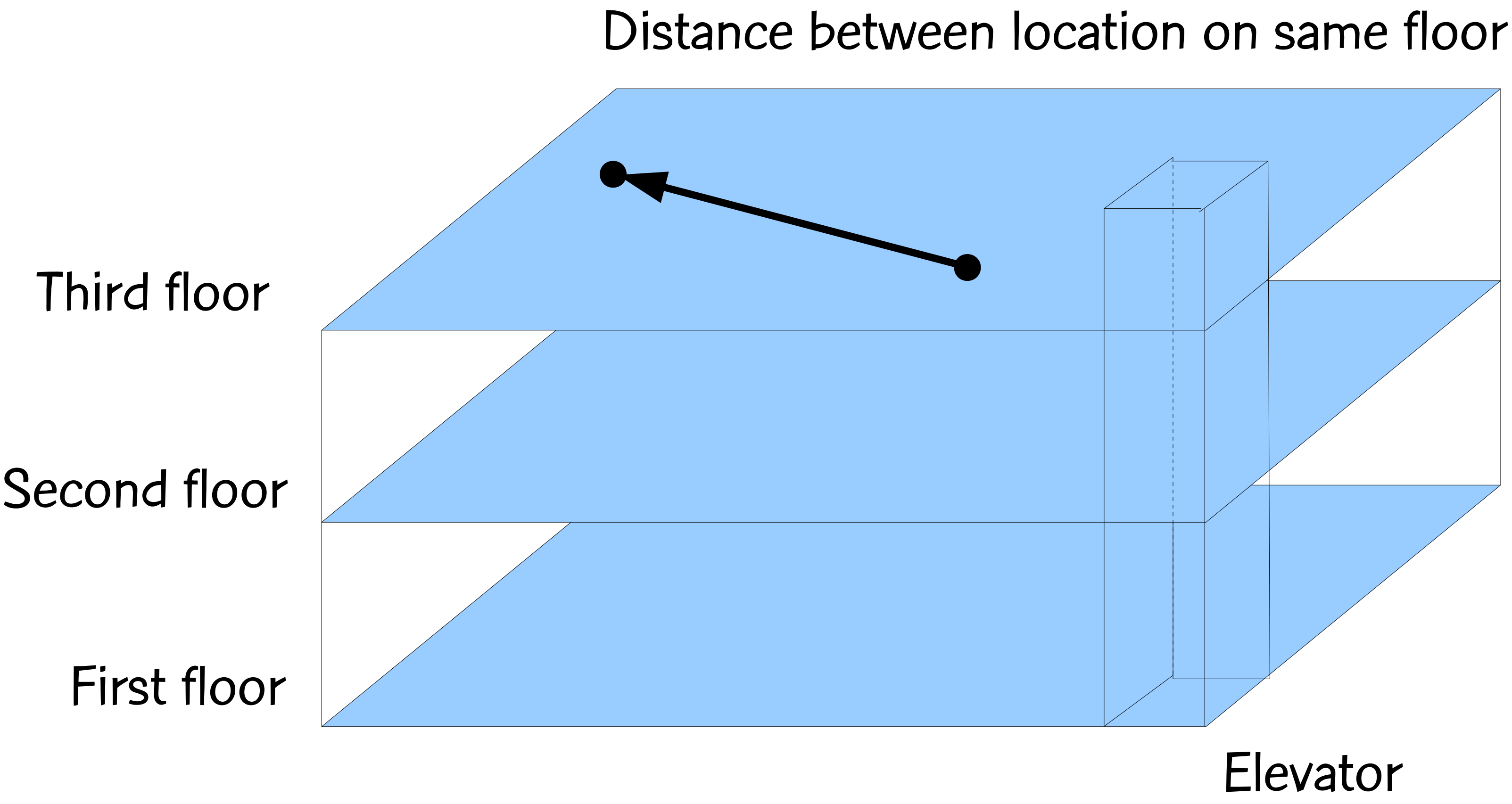
Third floor



Hospital layout as a GQAP

- Distances between locations on same floor are just the Euclidean distances between the centers of the locations.
- Distances between locations on different floors are the sums of the Euclidean distance between the center of the the first location to the elevator on that floor, the distance traveled by elevator (penalized), and the Euclidean distance between the elevator on the other floor and the center of the second location.

Hospital layout as a GQAP



Distance between location on same floor

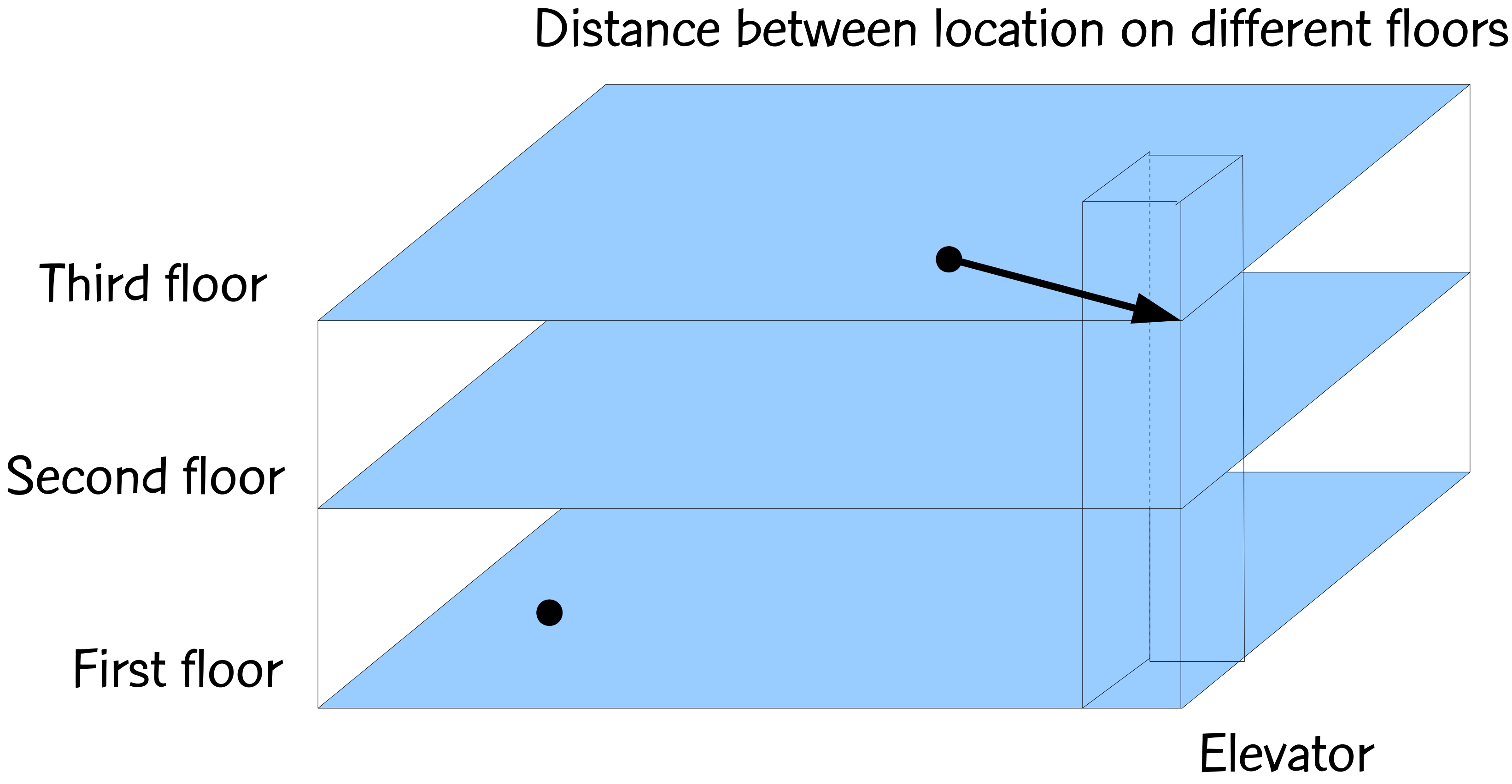
Third floor

Second floor

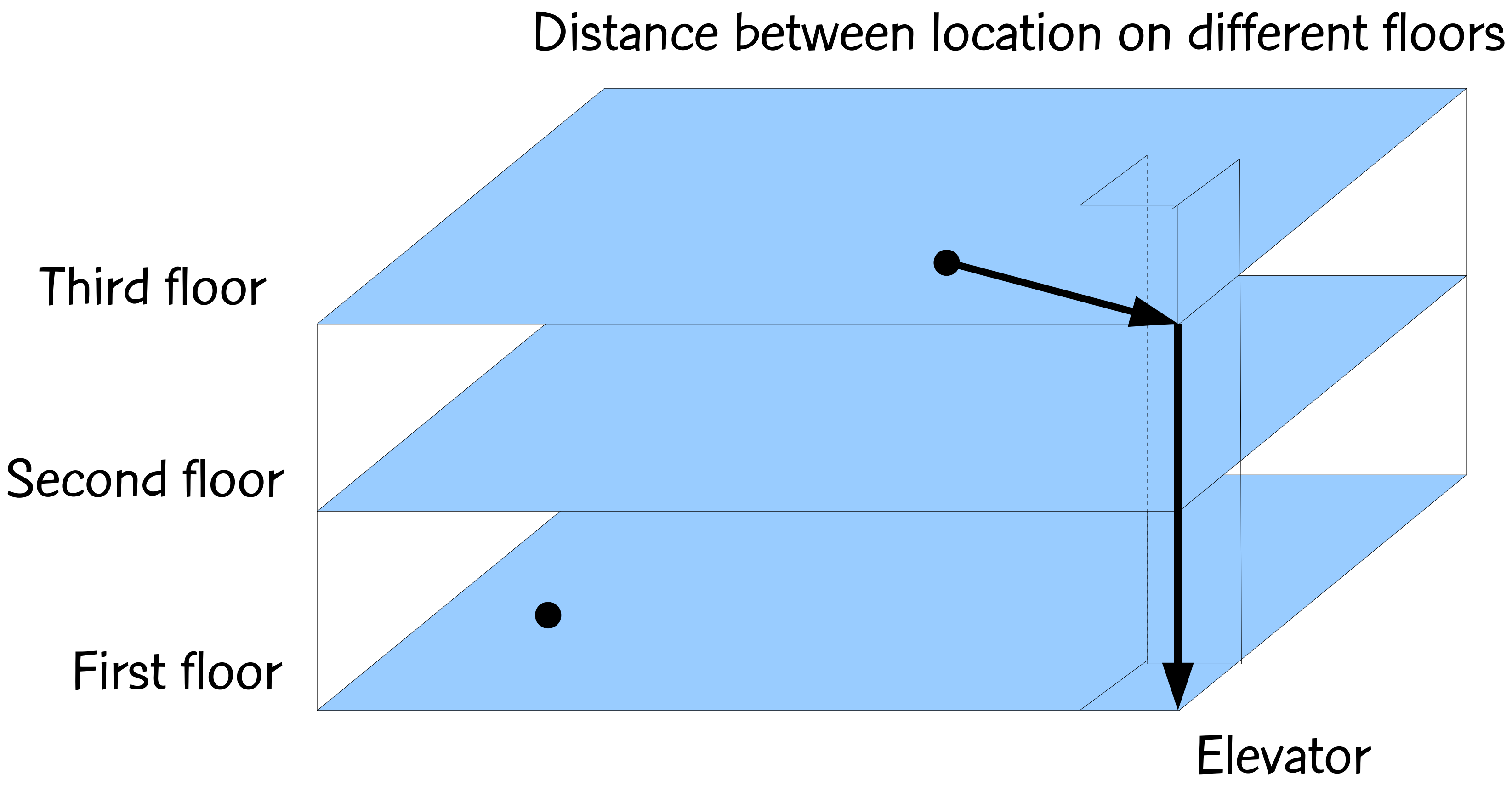
First floor

Elevator

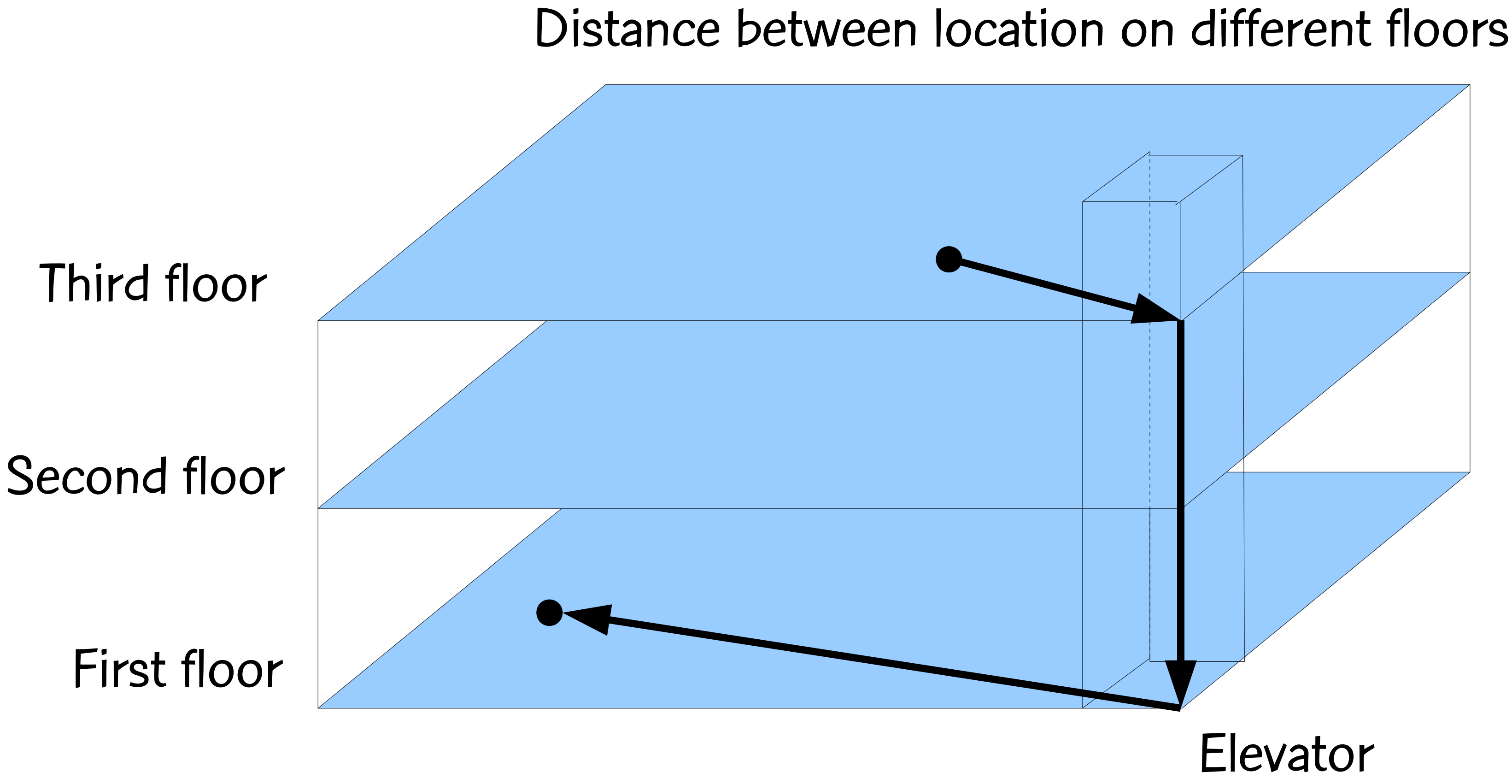
Hospital layout as a GQAP



Hospital layout as a GQAP



Hospital layout as a GQAP



Hospital layout as a GQAP

🟩 ICU ::: quantity: 3 ::: area 135 m^2

🟥 Pediatric ICU ::: quantity: 6 ::: area 110 m^2

🟨 Operating room ::: quantity: 12 ::: area 90 m^2

🟩 Radiology ::: quantity: 12 ::: area 65 m^2

🟪 Physician's office ::: quantity: 15 ::: area 45 m^2

Hospital layout as a GQAP

Inter-facility traffic is given

🟩 ICU ::: quantity: 3 ::: area 135 m^2

🟥 Pediatric ICU ::: quantity: 6 ::: area 110 m^2

🟨 Operating room ::: quantity: 12 ::: area 90 m^2

🟩 Radiology ::: quantity: 12 ::: area 65 m^2

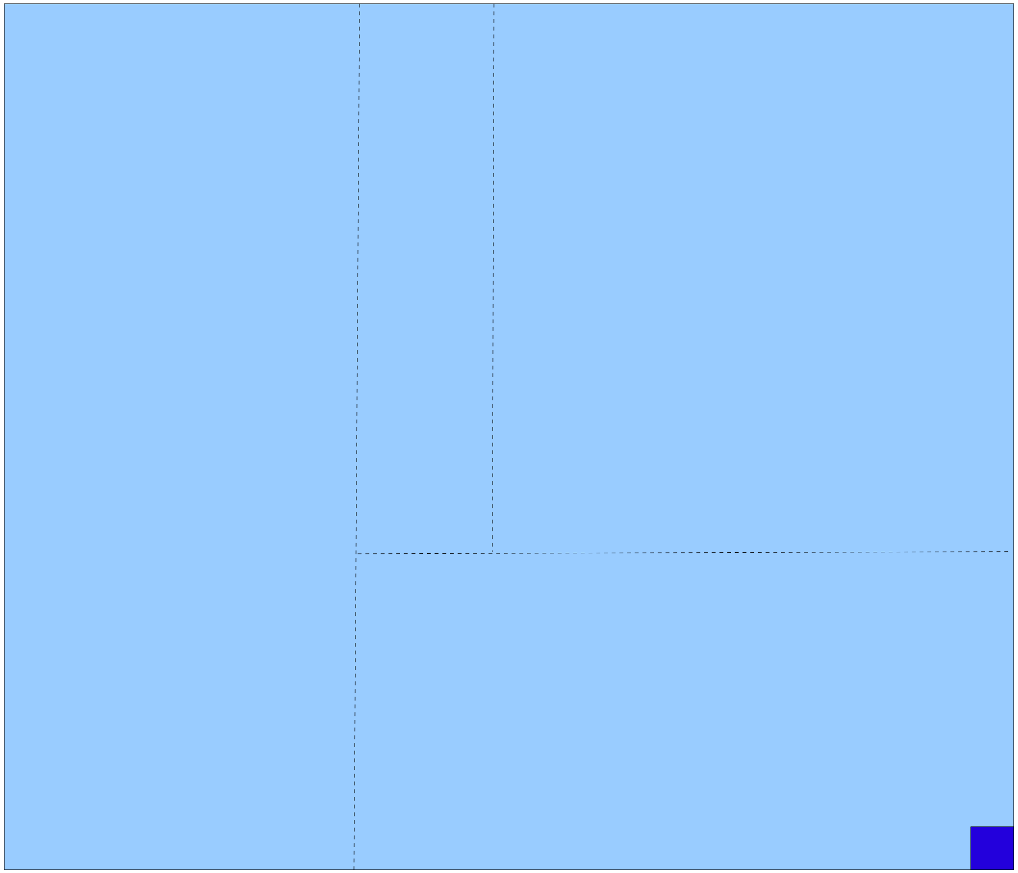
🟪 Physician's office ::: quantity: 15 ::: area 45 m^2

Hospital layout as a GQAP

- Applying GRASP with path-relinking heuristic, the following assignment was found in 1898.4 secs on a 2.6 Ghz machine.

Hospital layout as a GQAP

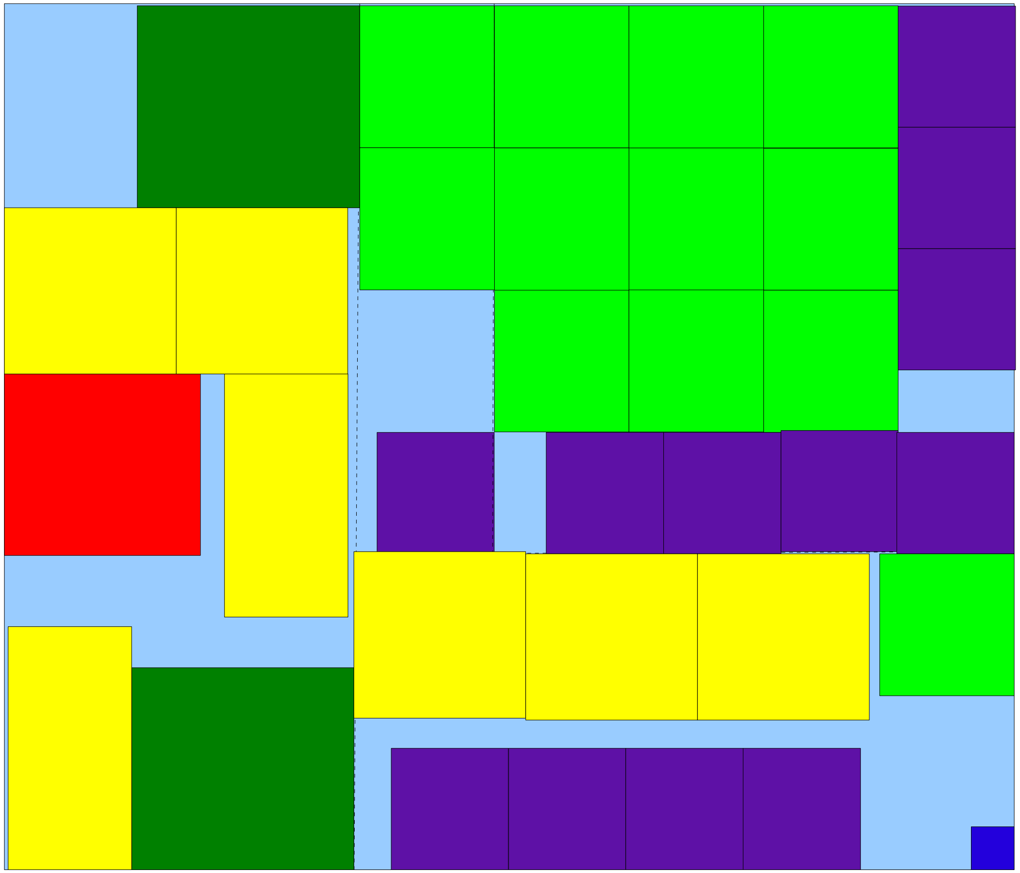
Third floor



Elevator

Hospital layout as a GQAP

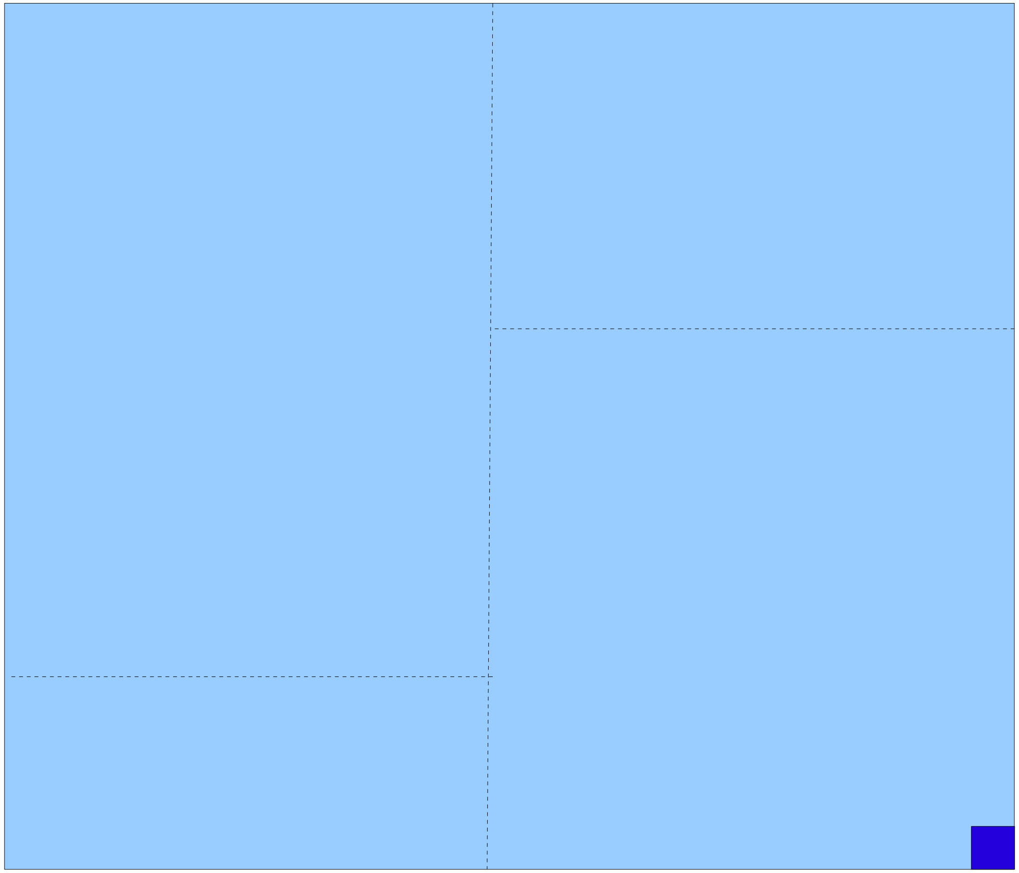
Third floor



Elevator

Hospital layout as a GQAP

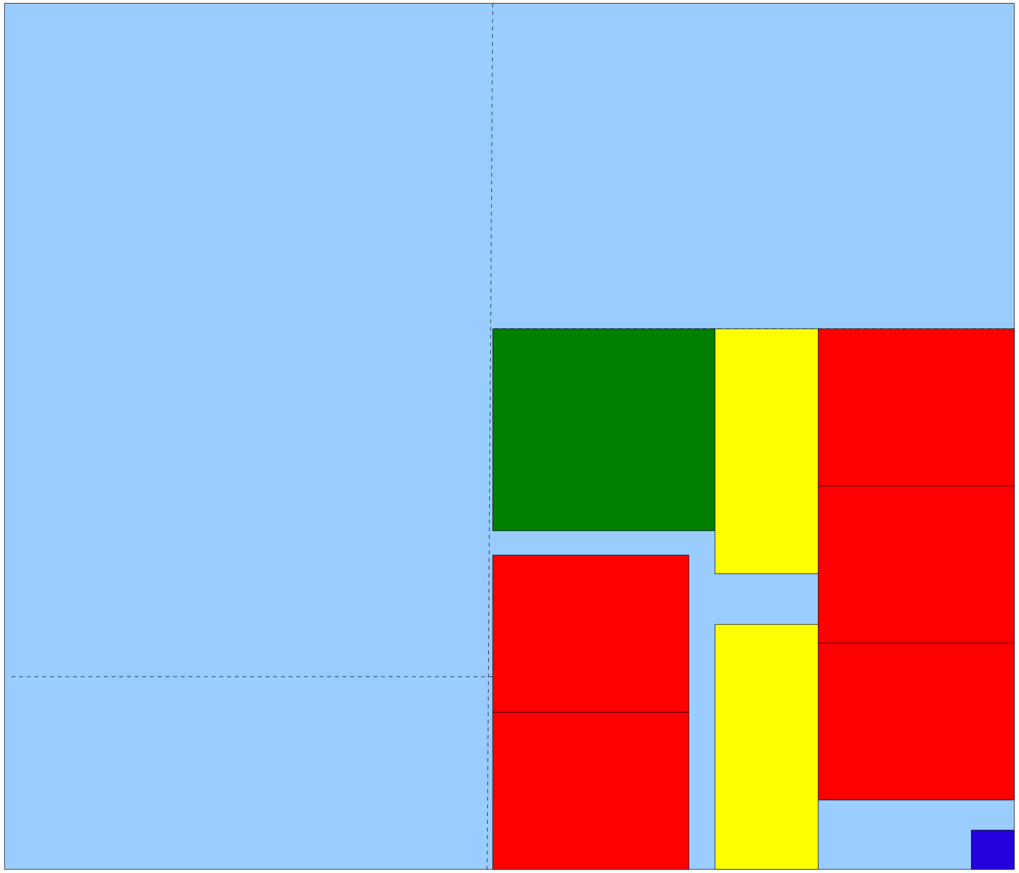
Second floor



Elevator

Hospital layout as a GQAP

Second floor



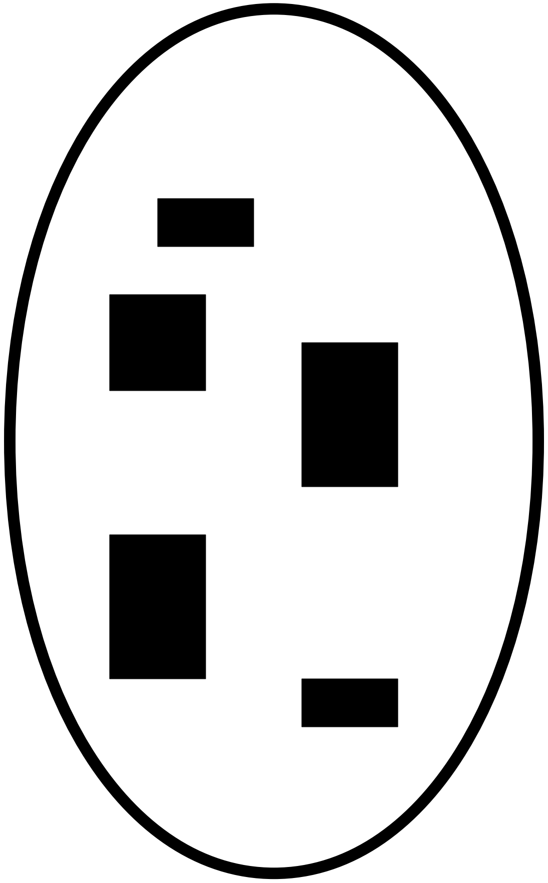
Elevator

Generalized quadratic assignment problem

Generalized quadratic assignment

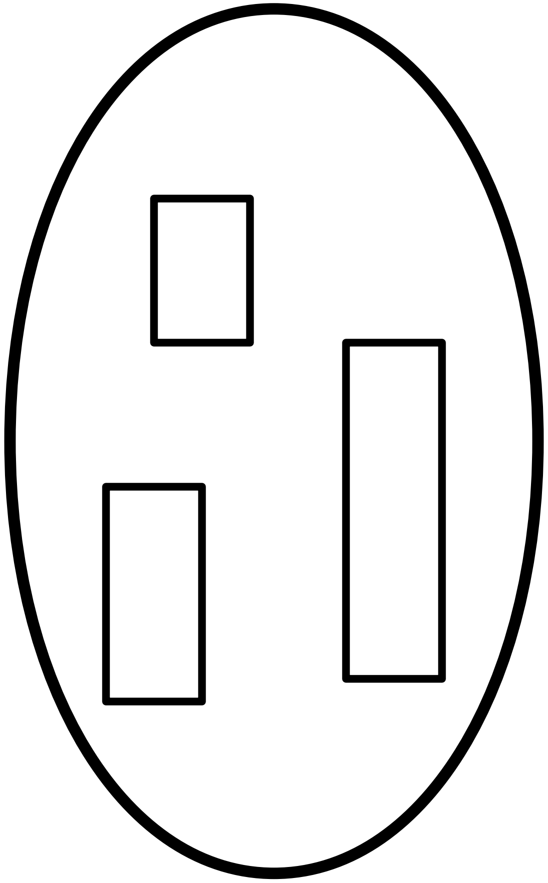
- The GQAP is NP-hard.
- It is a generalization of the quadratic assignment problem (QAP).
- Multiple facilities can be assigned to a single location as long as the capacity of the location allows.

N: set of n facilities



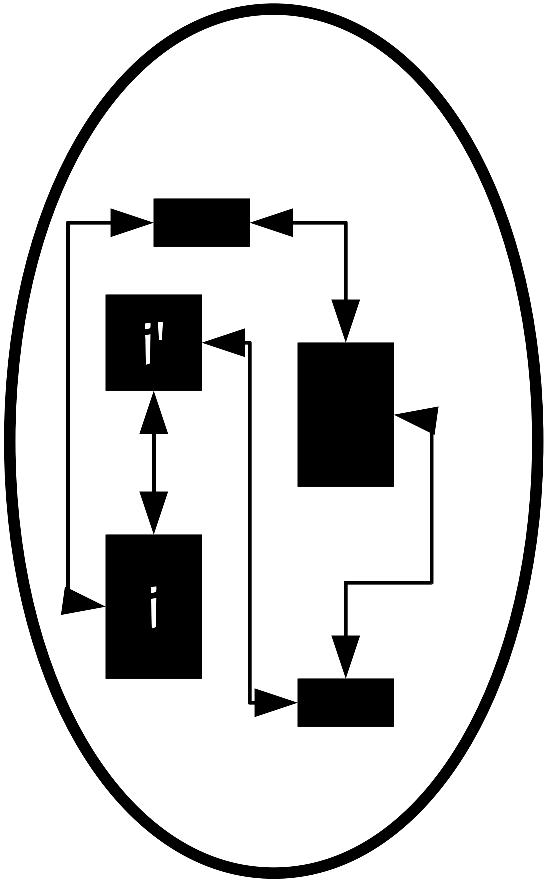
d_i : capacity demanded by facility $i \in N$

M: set of m locations

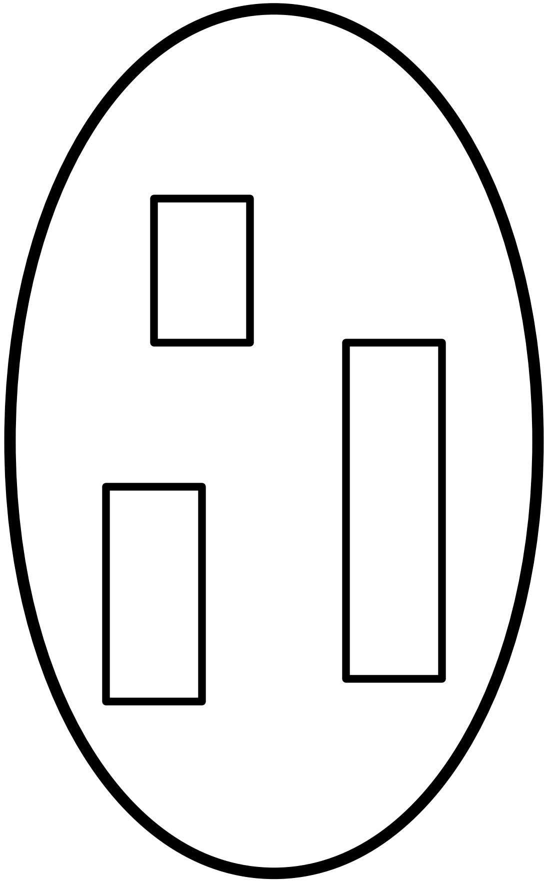


Q_j : capacity of location $j \in M$

N: set of n facilities



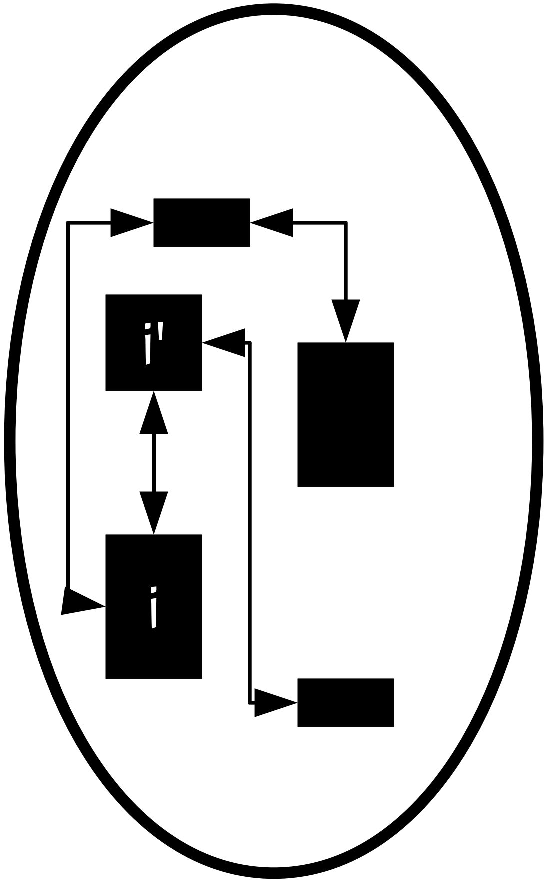
M: set of m locations

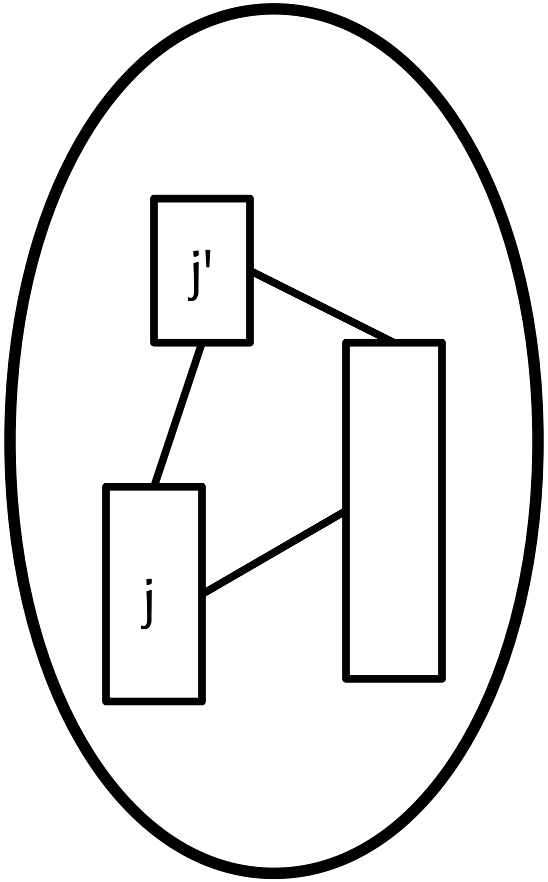


$A_{nxn}=(a_{ii'})$: flow between facilities

N: set of n facilities

M: set of m locations



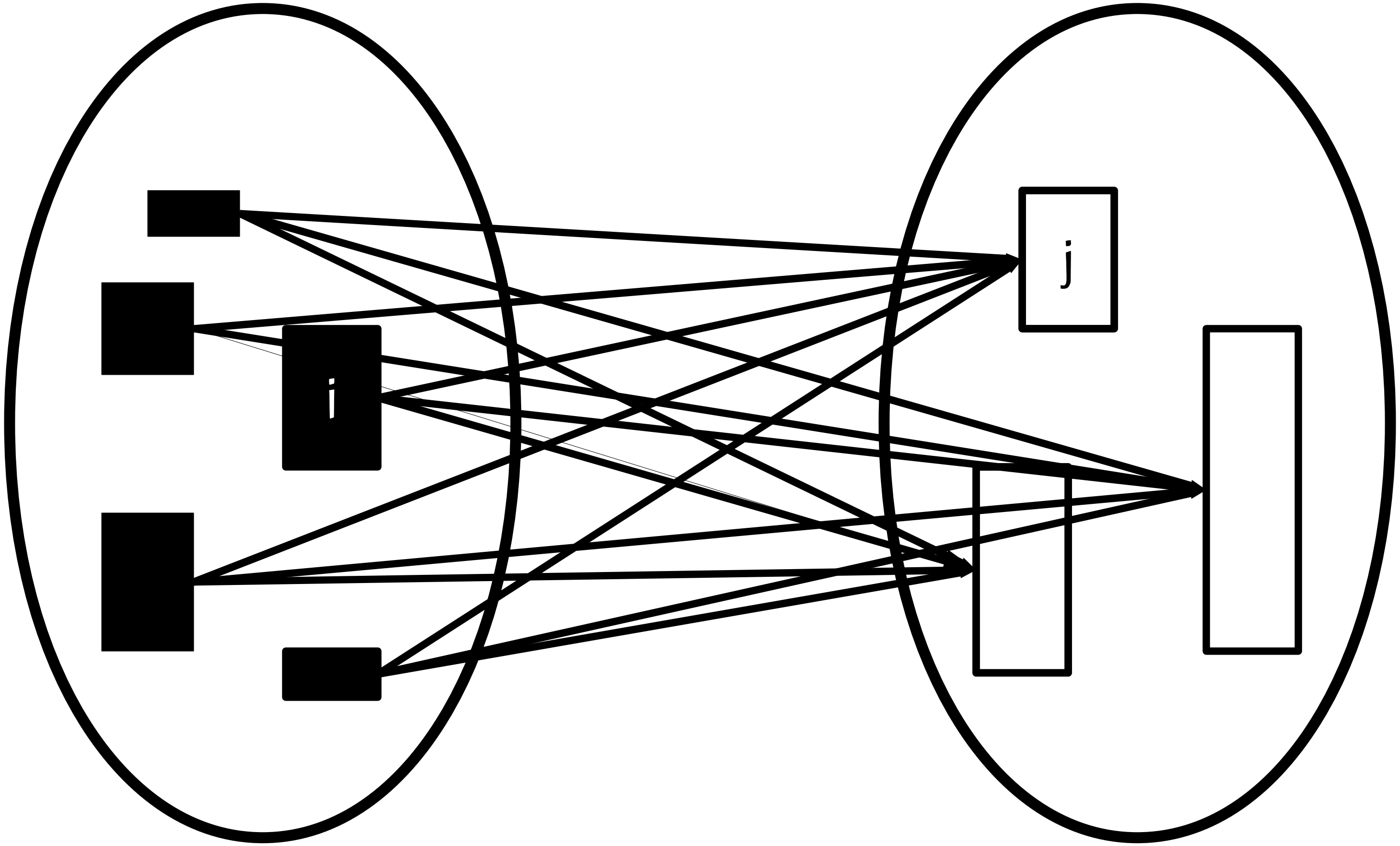


$A_{nxn} = (a_{ii'})$: flow between facilities

$B_{mxm} = (b_{jj'})$: distance between locations

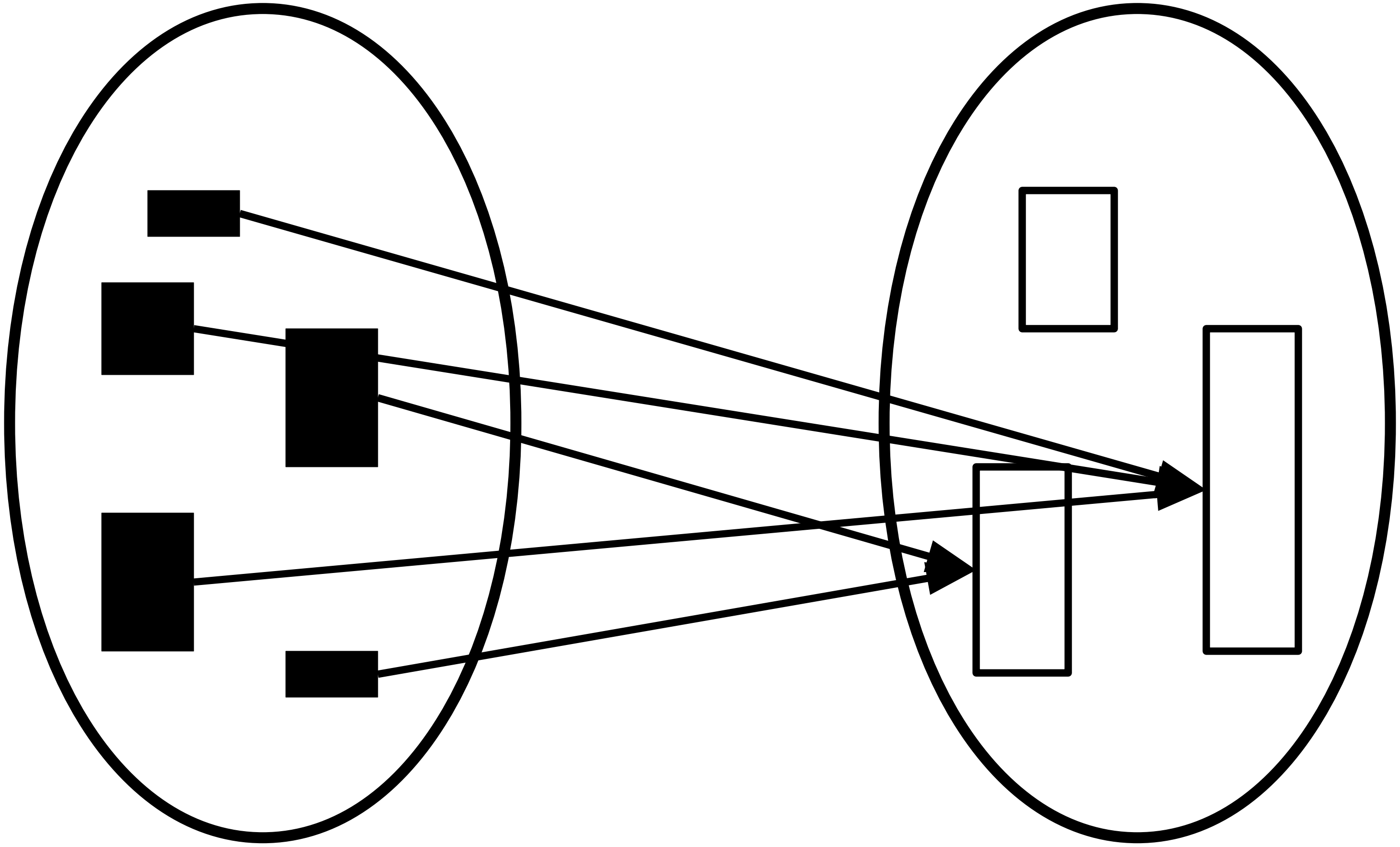
N: set of n facilities

M: set of m locations



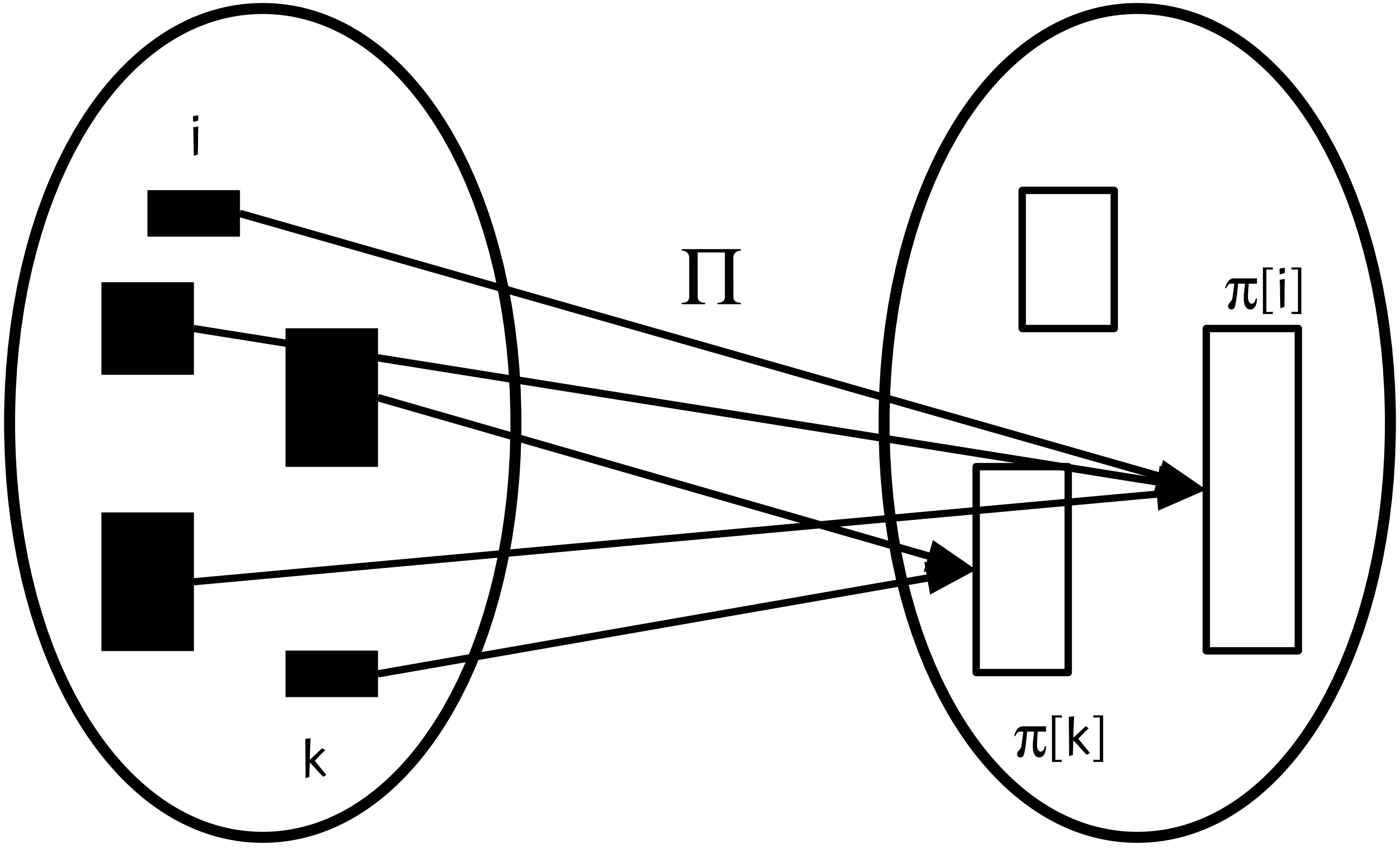
$C_{nxm}=(c_{ij})$: cost of assigning facility i∈N to location j∈M

The generalized quadratic assignment problem



GQAP seeks a assignment, without violating the capacities of locations, that minimizes the sum of products of flows and distances in addition to a linear total cost of assignment.

The generalized quadratic assignment problem

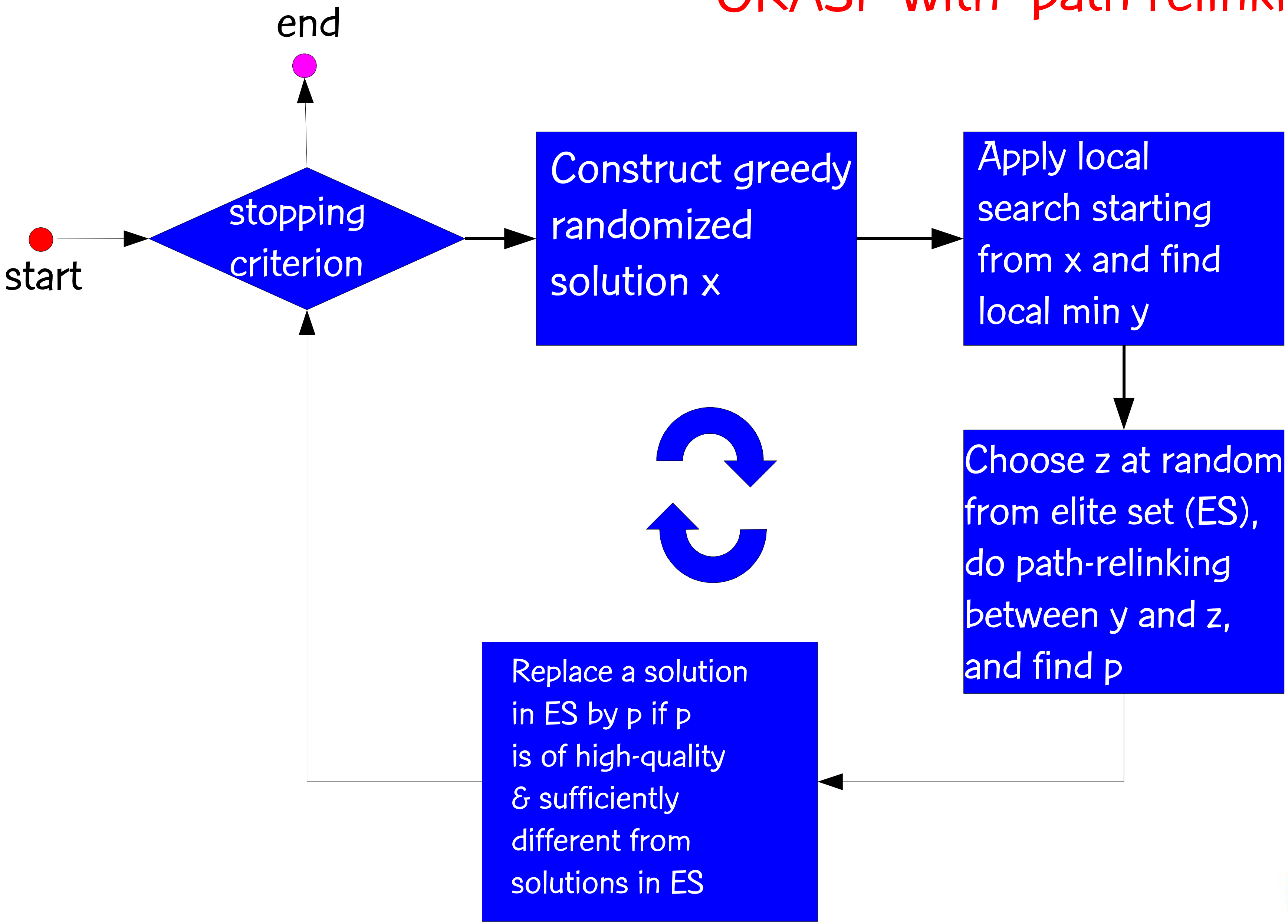


cost[Π] = sum(i=1,n) c[i,π[i]] +

sum(i=1,n) sum (i≠k=1,n) F[i,k]*D[π[i],π[k]]

Solution method

GRASP with path-relinking



end

start

stopping criterion

Construct greedy randomized solution x

Apply local search starting from x and find local min y

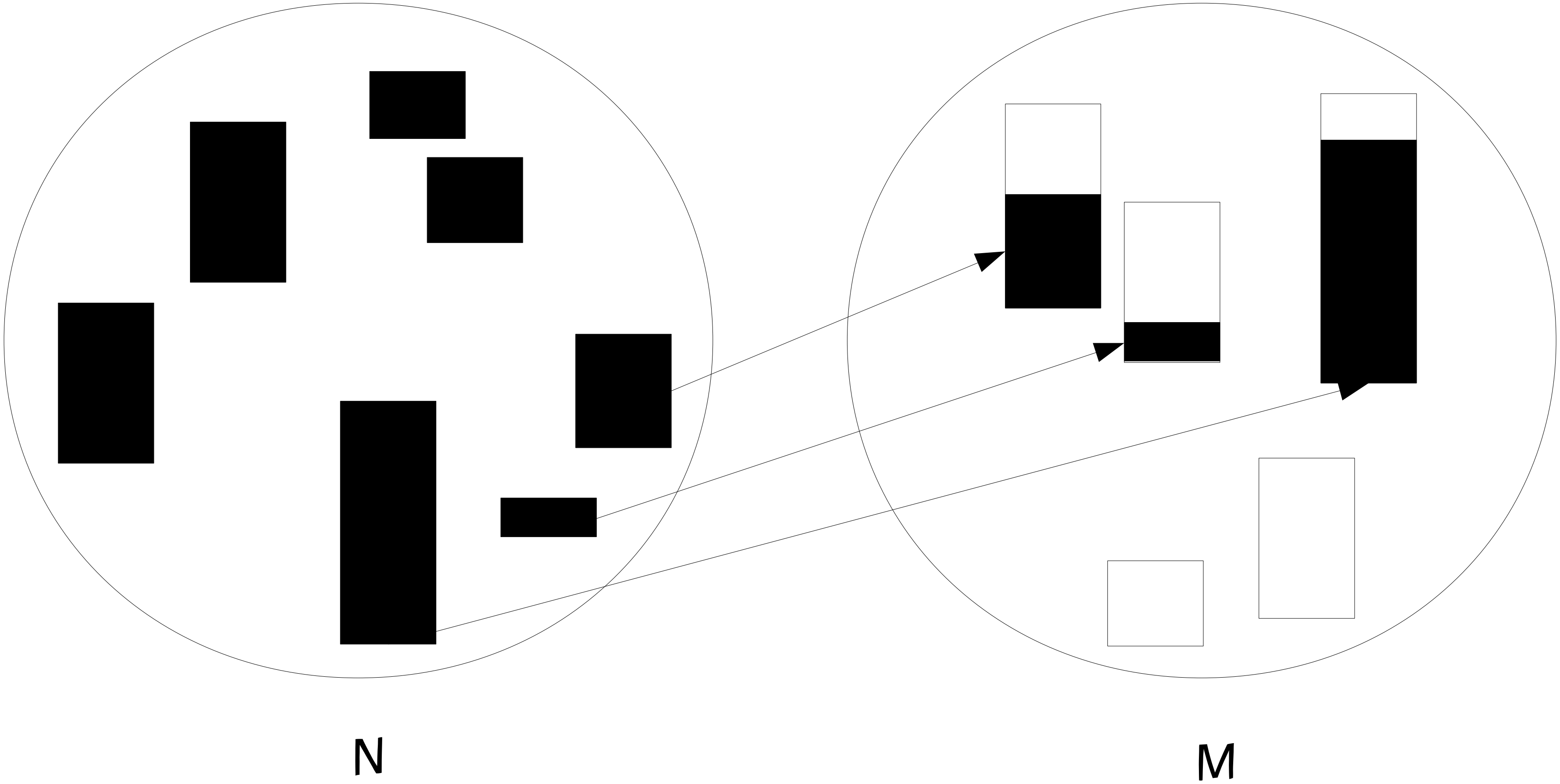
Choose z at random from elite set (ES), do path-relinking between y and z, and find p

Replace a solution in ES by p if p is of high-quality & sufficiently different from solutions in ES

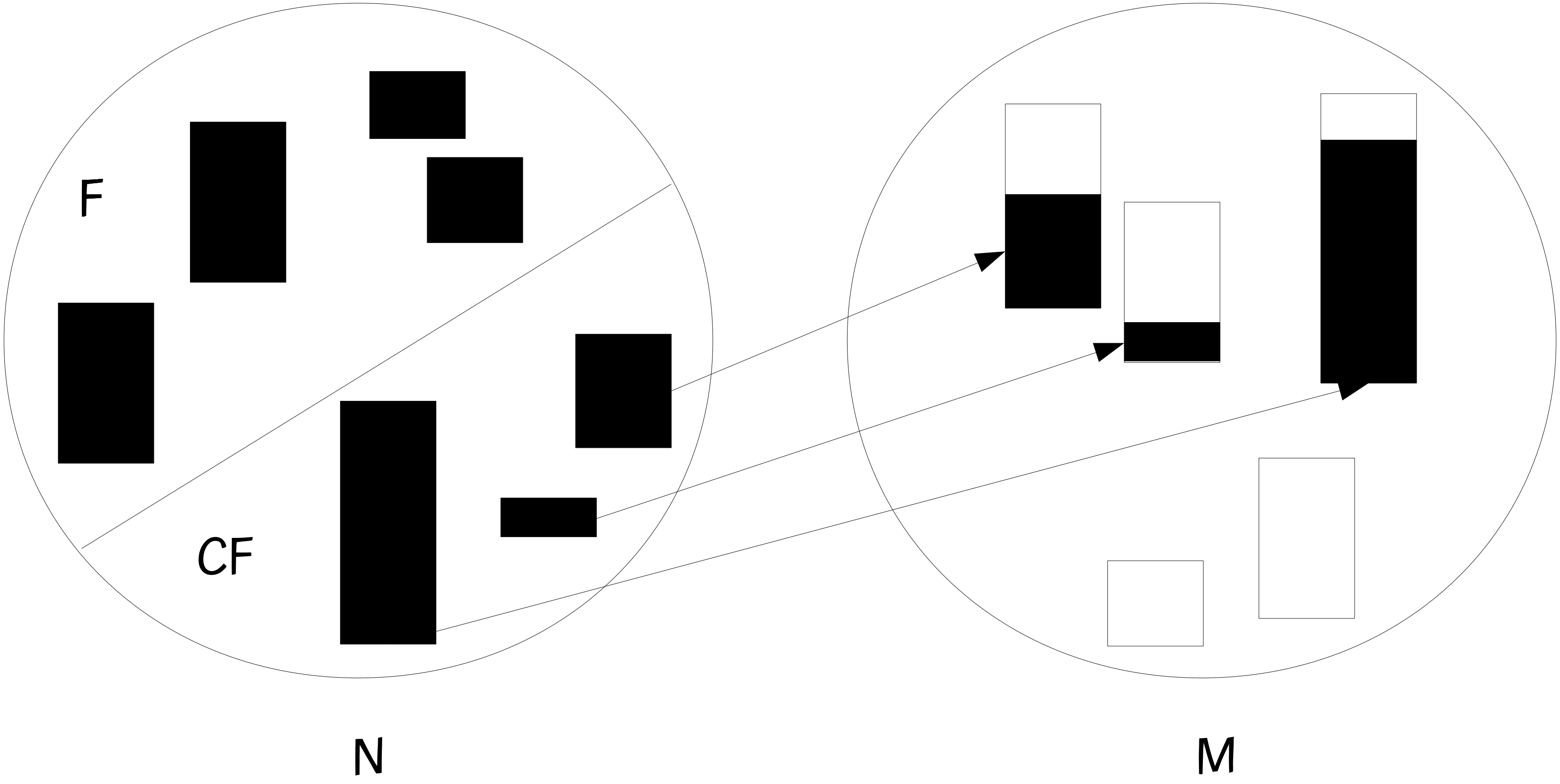
Components

- Construction of greedy randomized solution
- Local search
- Path-relinking

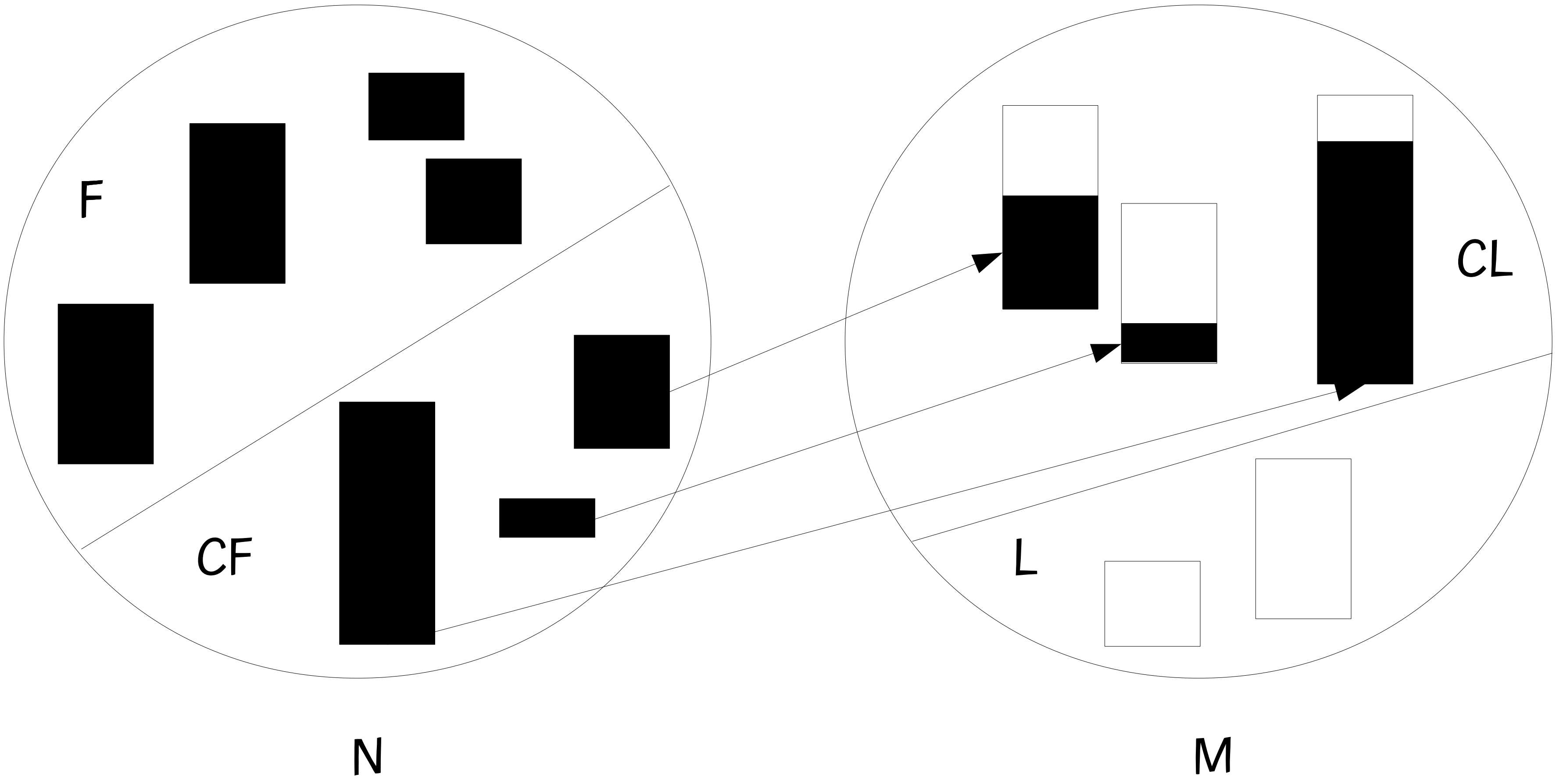
GRASP construction



Suppose a number of assignments have already been made

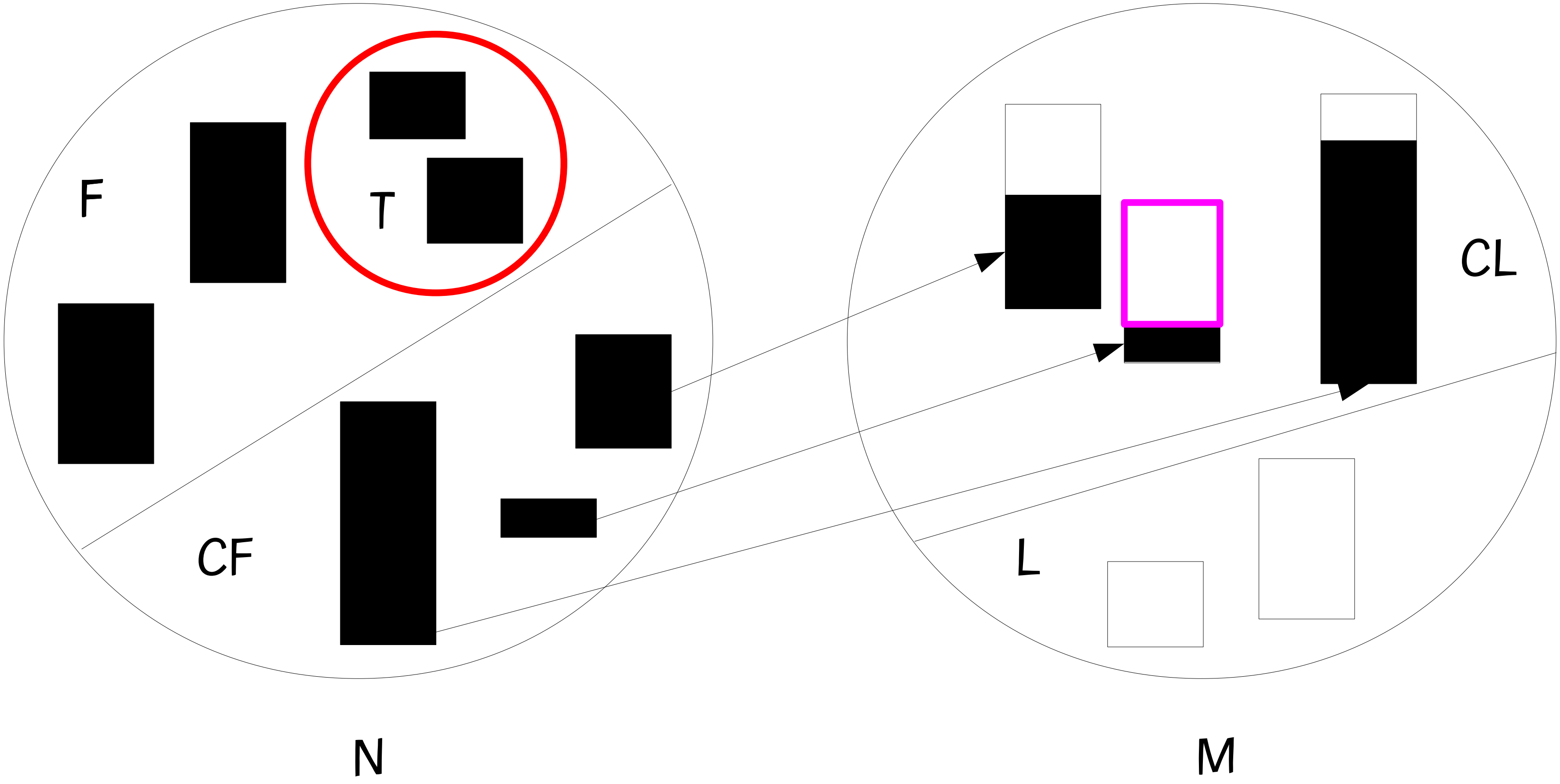


$N = F \cup CF$, where CF is the set of assigned facilities and F the set of facilities not yet assigned to some location



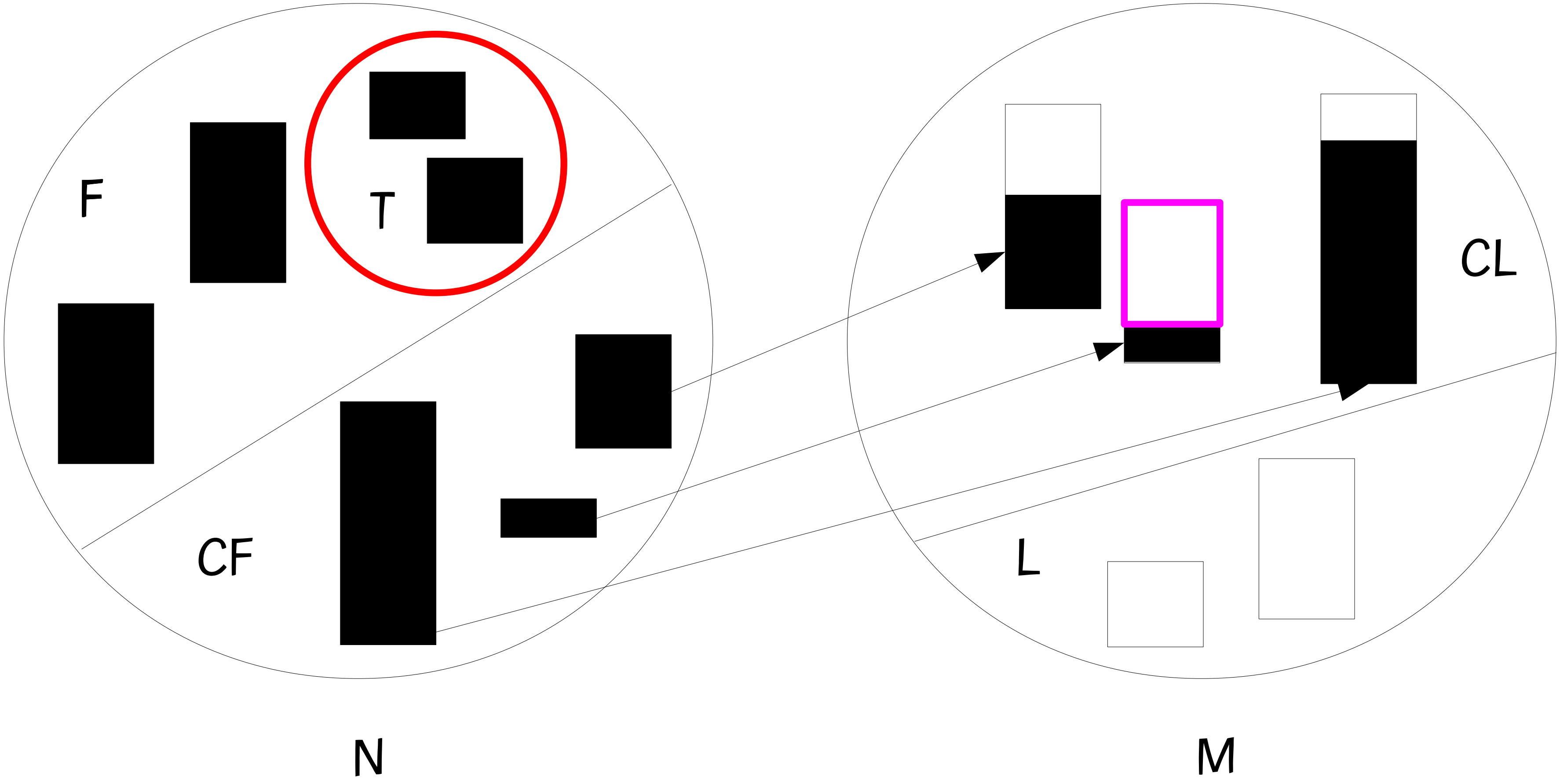
$M = L \cup CL$, where CL is the set of previously chosen locations and L the set of unselected locations.

Procedure to select a new location from set L



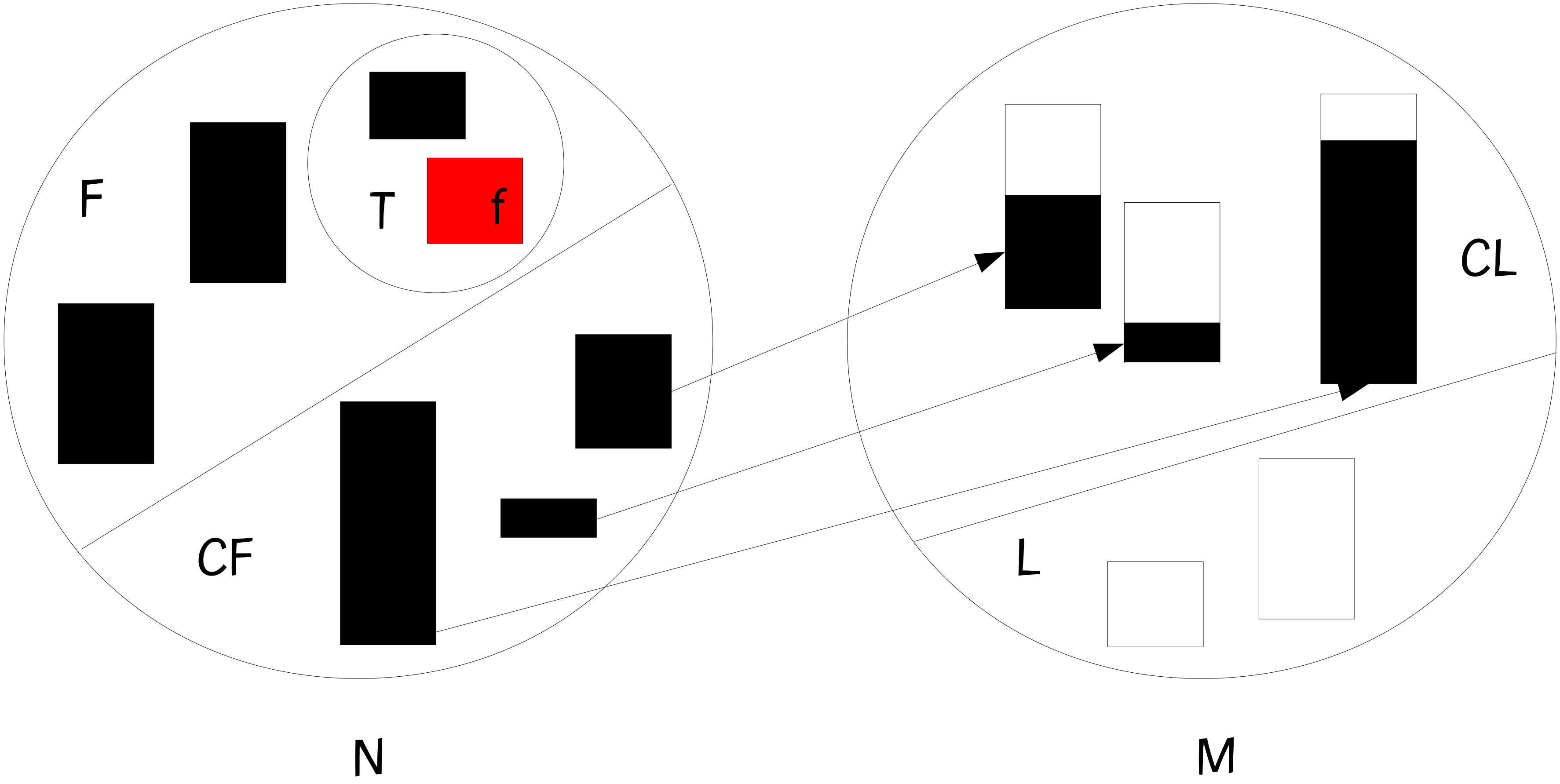
With probability 1– (|T|/|F|), randomly select a new location l from L, where the set T consists of all unassigned facilities with demands less than or equal to the maximum available capacity of locations in CL and move location l to CL

Procedure to select a new location from set L



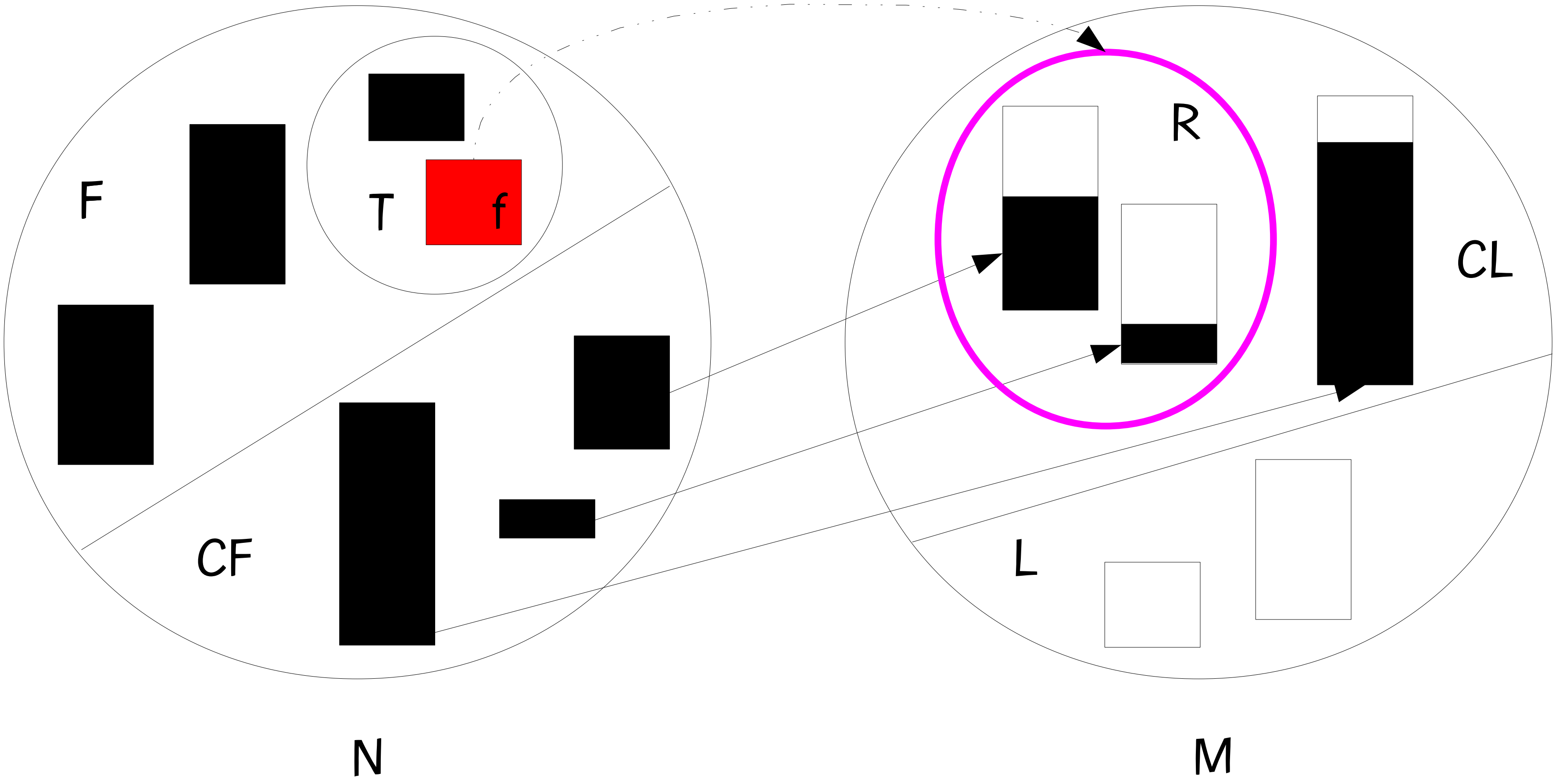
Favor locations in L that have high available capacity and that are close to all locations in CL

Facility selection procedure



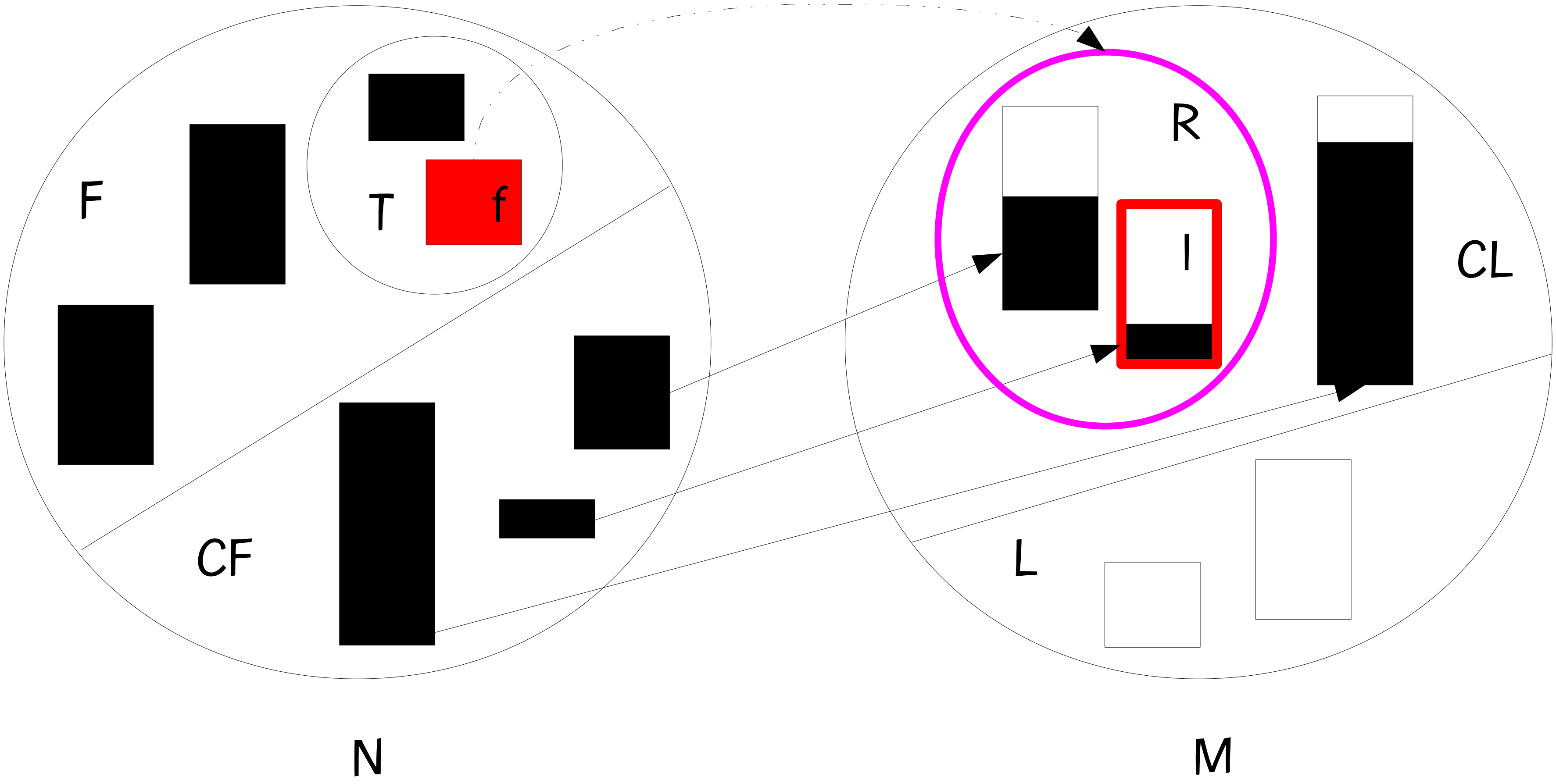
Randomly select a facility f ∈ T favoring facilities that have high demand and high flows to other facilities.

Procedure to select a location from CL (step 1)



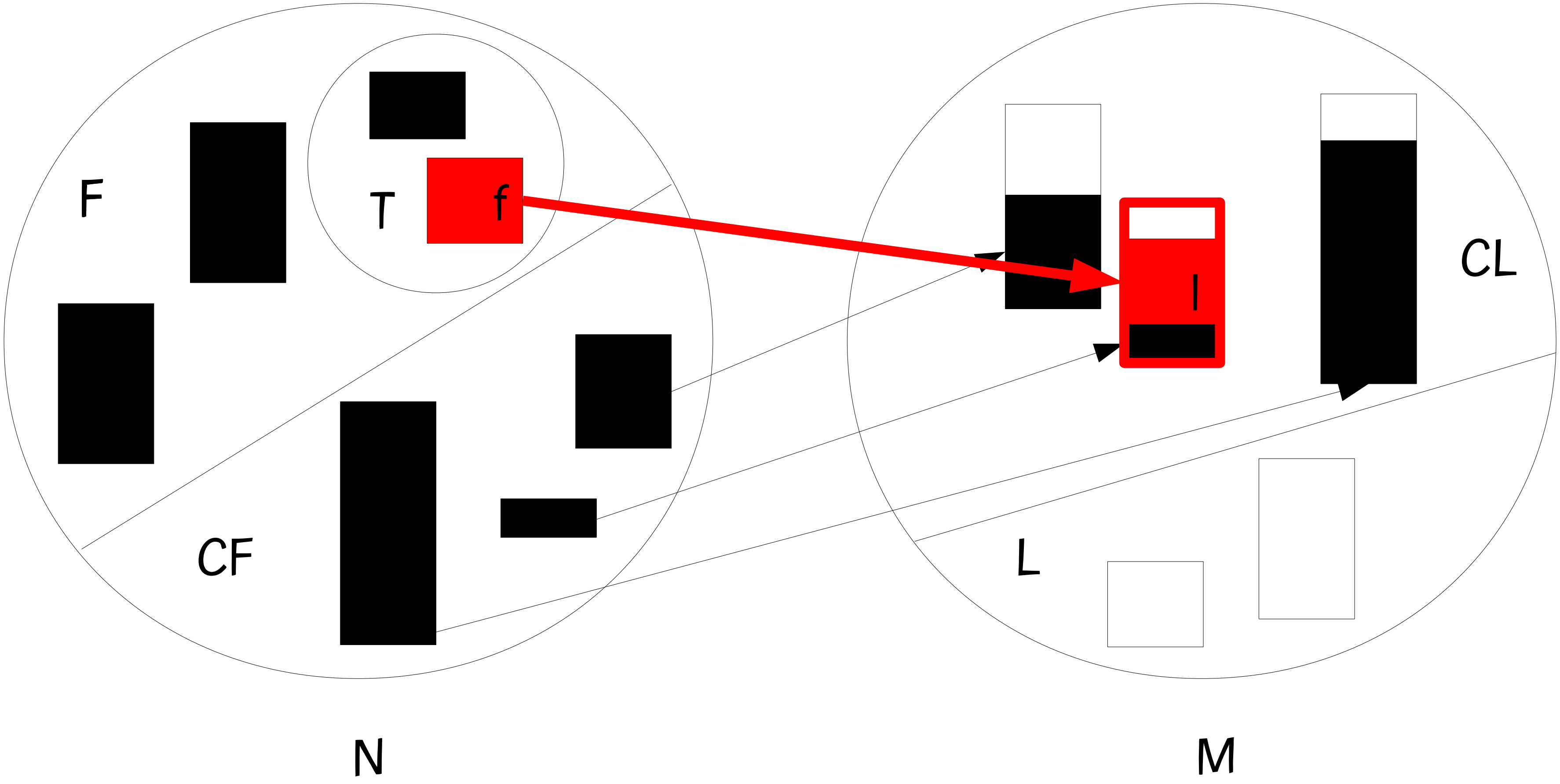
1. Let set R to be all locations in CL having slack greater than or equal to demand of facility f;

Procedure to select a location from CL (step 2)



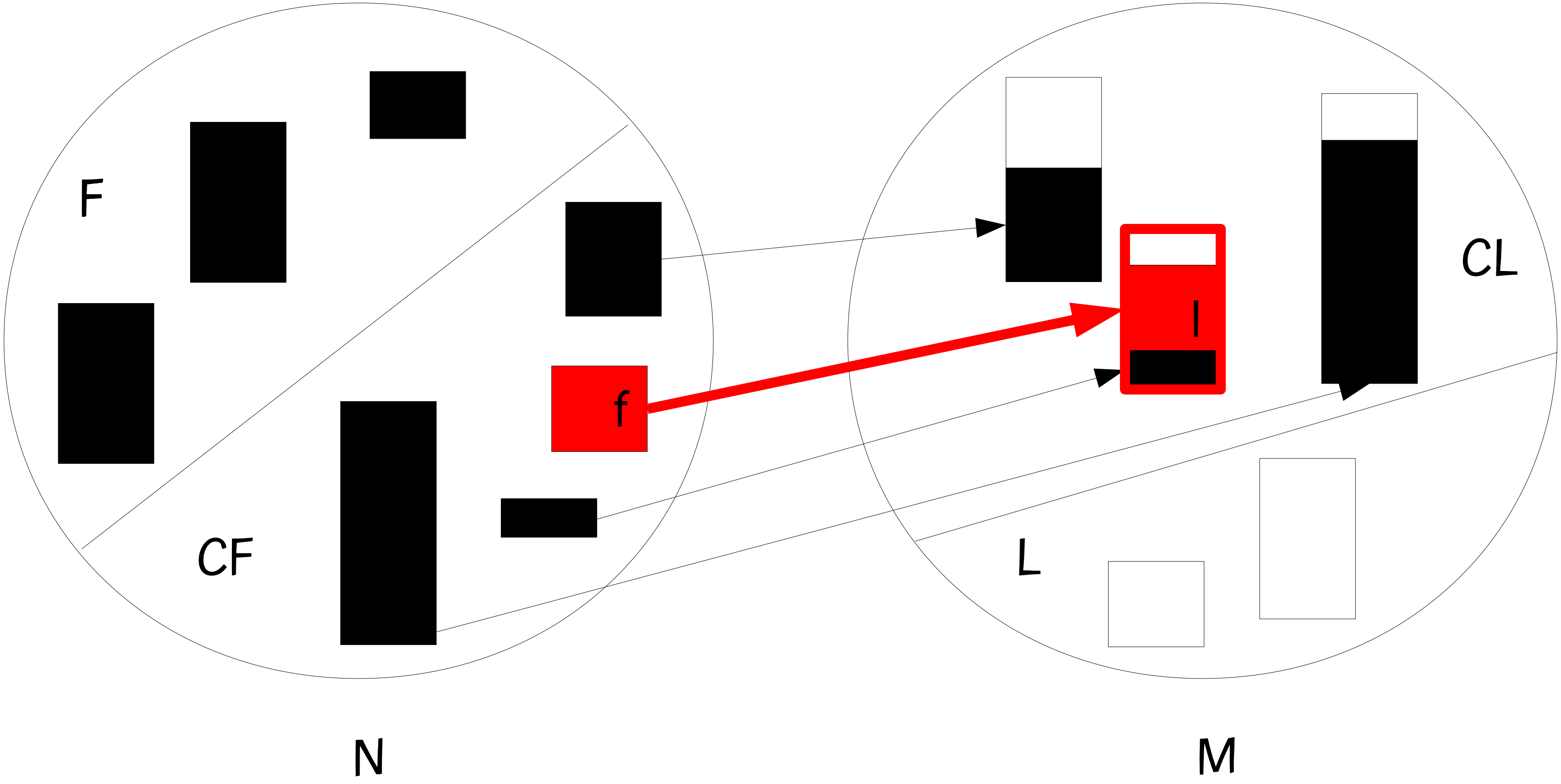
2. Randomly select a location l ∈ R favoring those having high available capacity and those close to high-capacity locations in CL;

Assignment procedure



Assign facility f to location l

Assignment procedure



Update sets F, CF, and slack of location l

Considerations about the construction procedure

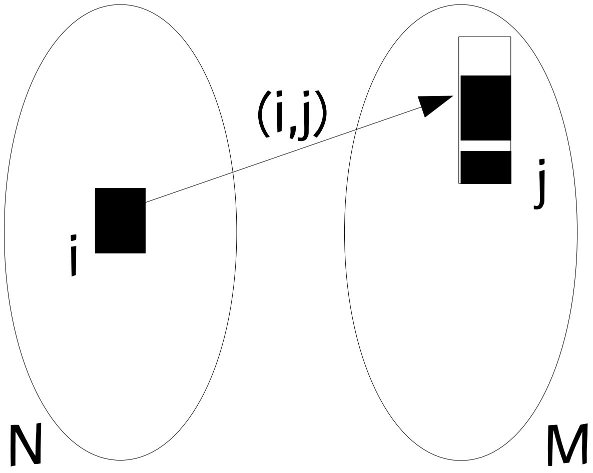
- The procedure is not guaranteed to produce a feasible solution.
- To address this difficulty, the construction procedure is repeated a maximum number of times or until all facilities are assigned (i.e. until $F=\emptyset$).
- At start of construction, a location l from L is selected with probability proportional to its capacity. Location l is placed in CL.

Local search

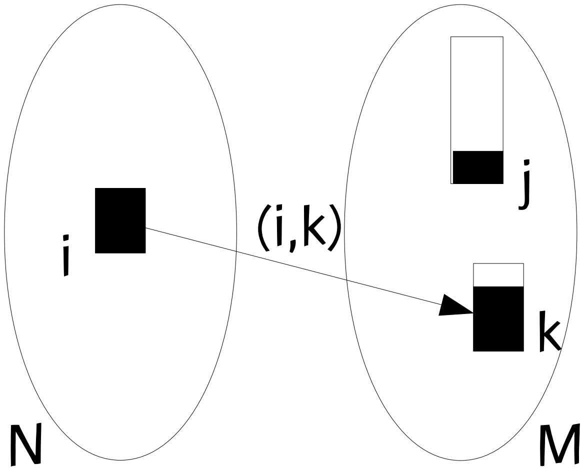
Local search

1-move and 2-move neighborhoods from solution p are used in our local search.

1-move: changing one facility-to-location assignment in p



solution p



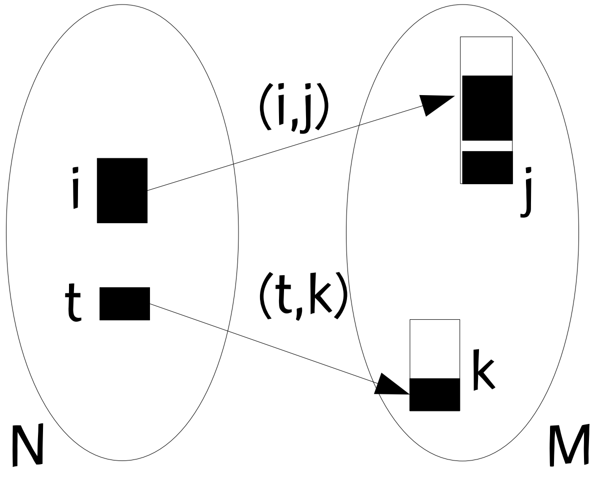
1-move neighbor of p

Local search

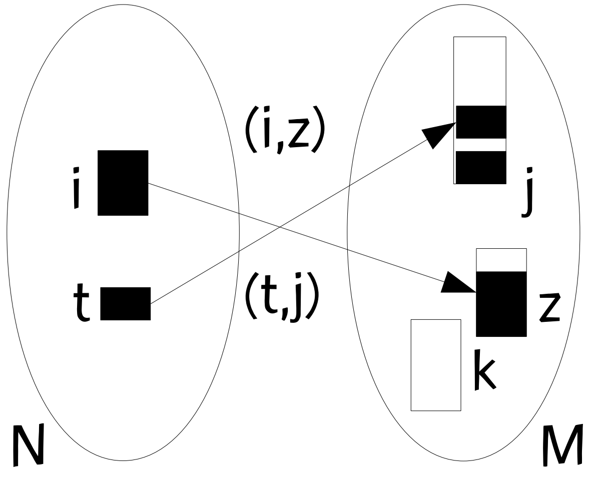
1-move and 2-move neighborhoods from solution p are used in our local search.

1-move: changing one facility-to-location assignment in p

2-move: changing two facility-to-location assignment in p.

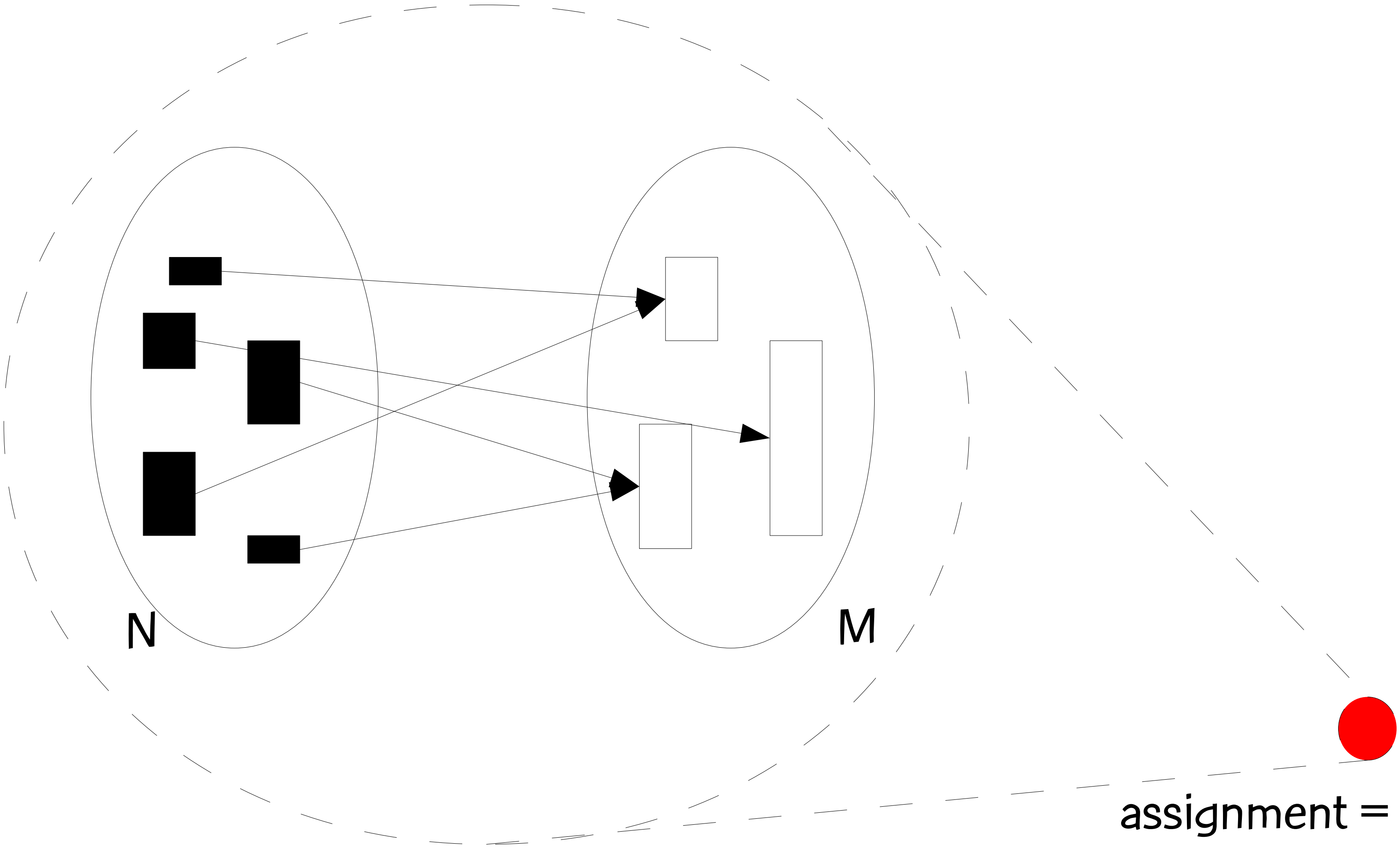


solution p



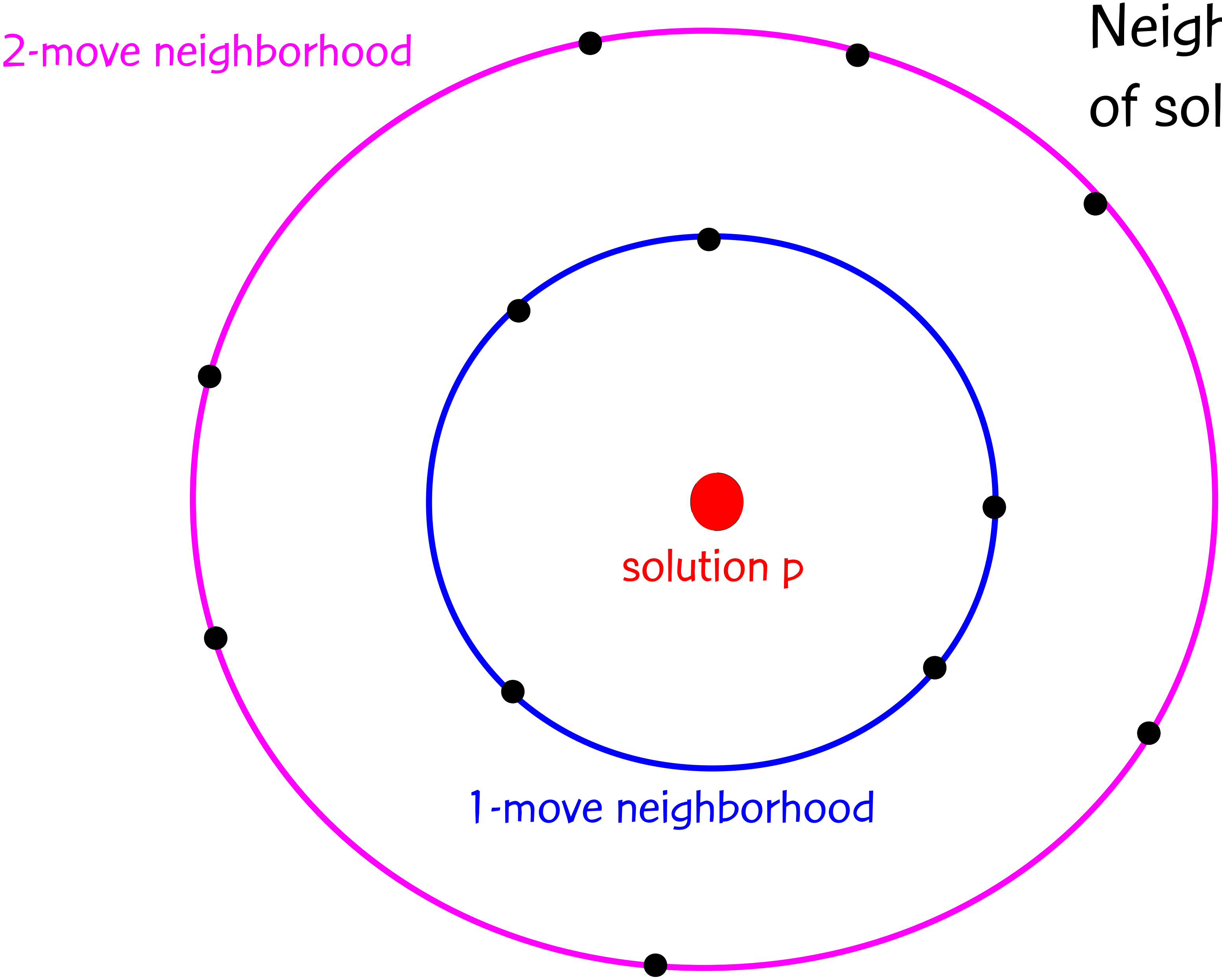
2-move neighbor of p

Assignment representation



assignment = solution

Neighborhood of solution p



2-move neighborhood

solution p

1-move neighborhood

Traditional local search approaches

Best improving approach:

Evaluate all 1-move and 2-move neighborhood solutions and select the best improving solution

First improving approach:

1: From solution p, to evaluate its 1-move neighbors until the first improving solution q is found.

2: If q does not exist, continue search in the 2-move neighborhood.

3: If q does not exist in the 2-move neighborhood, stop. Otherwise, assign p = q and go to step 1.

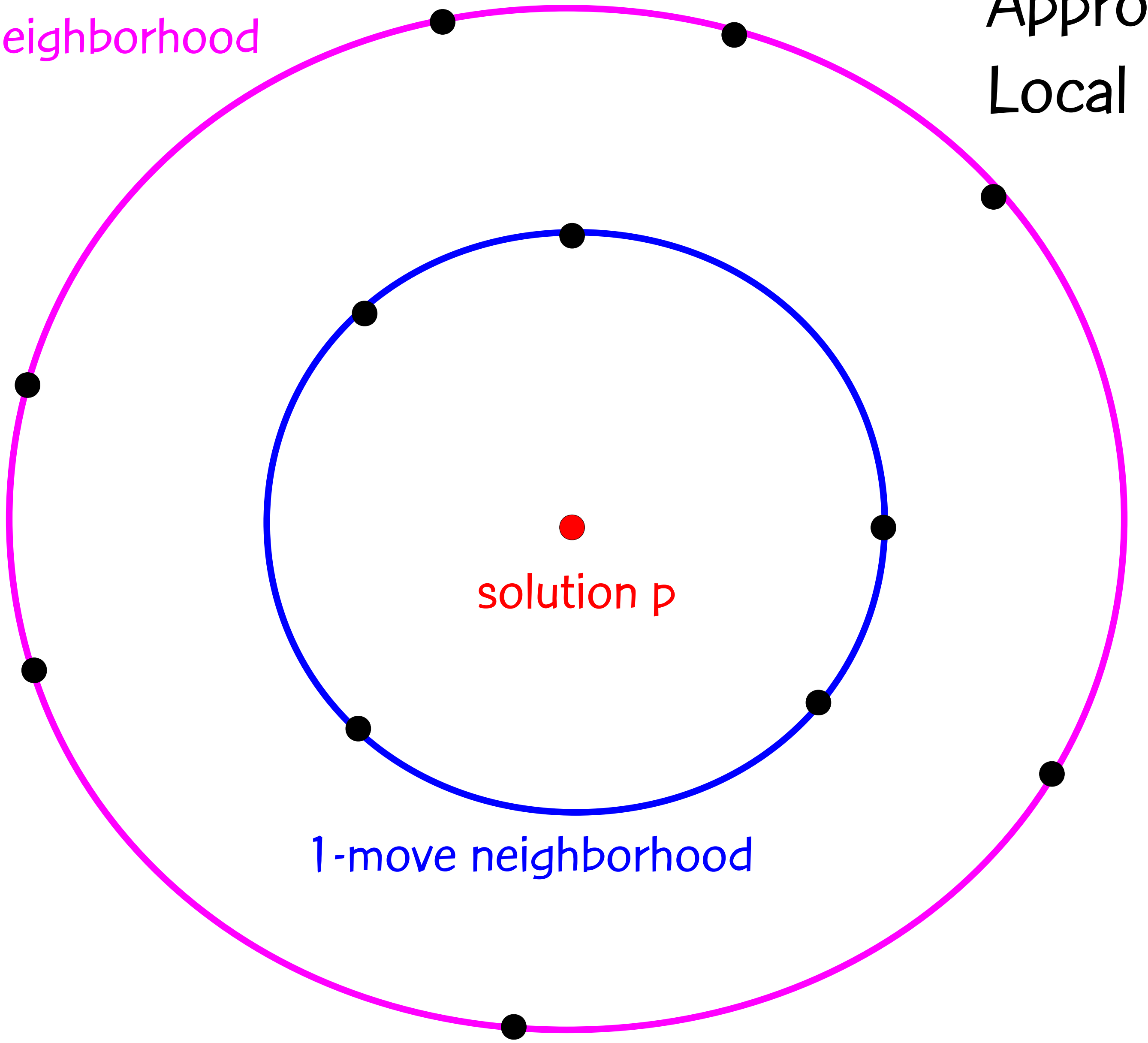
Approximate local search

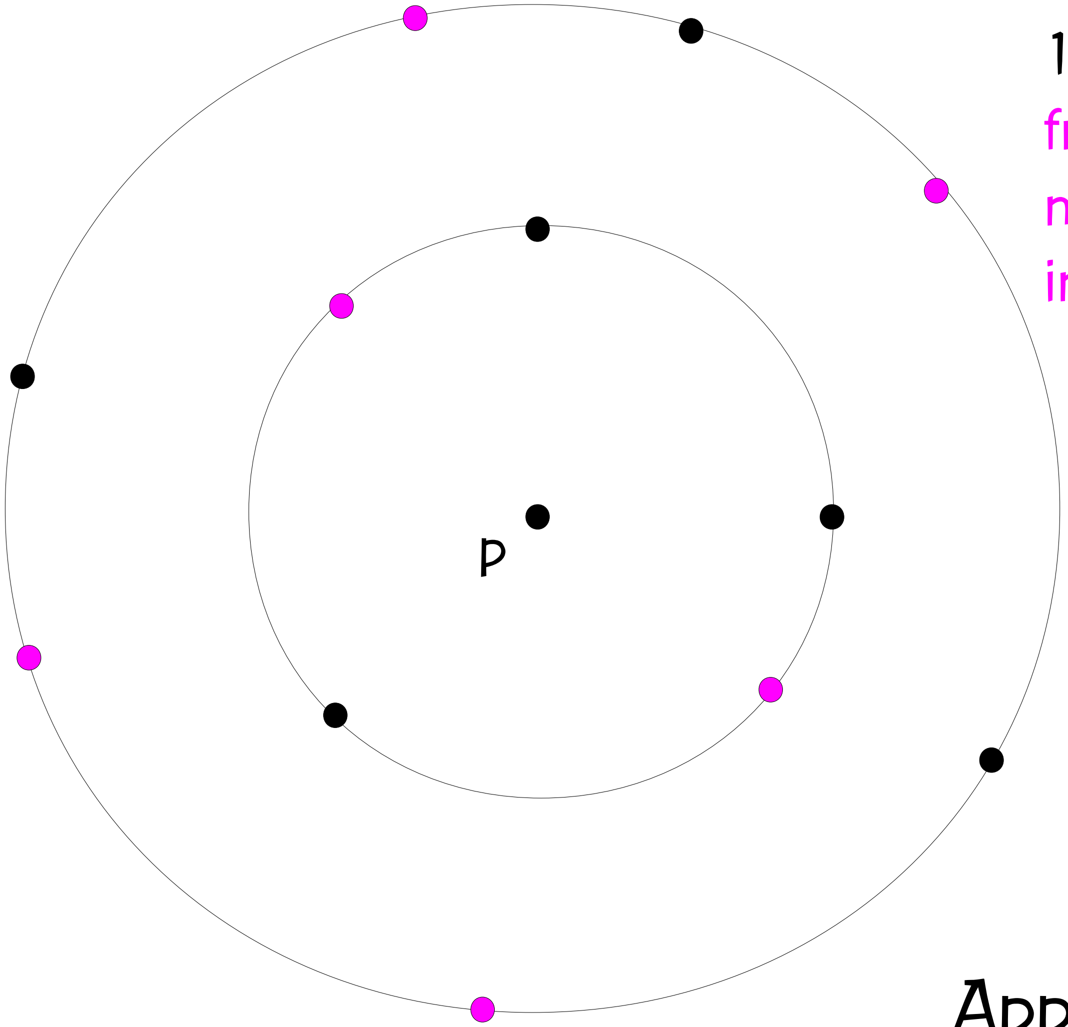
Neighborhoods can be very large for best improvement

Local search can take very long

Tradeoff between best & first improvement: sample the neighborhood of solution p.

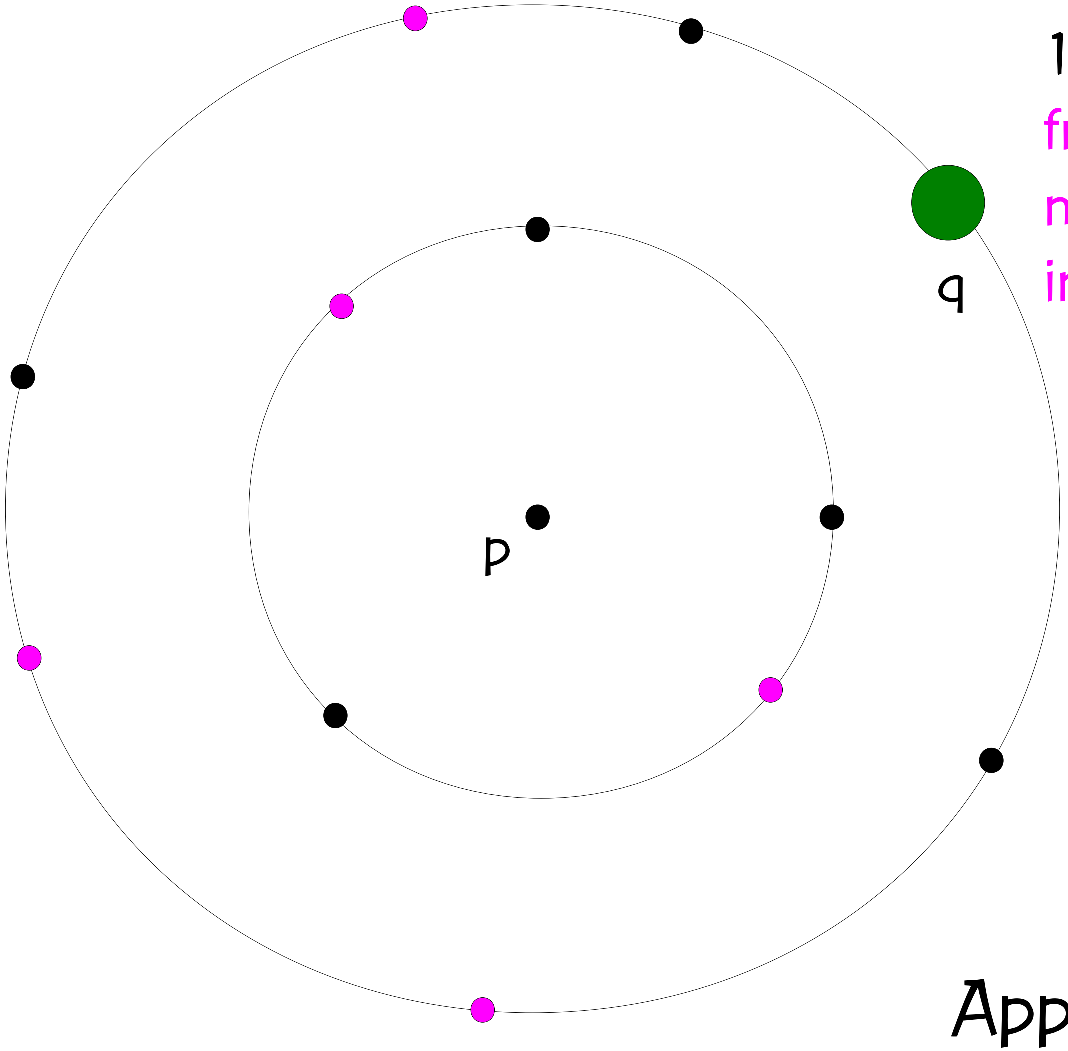
Approximate Local Search





1. Sample k improving solutions from 1-move and 2-move neighborhood of p and place them in an elite set E.

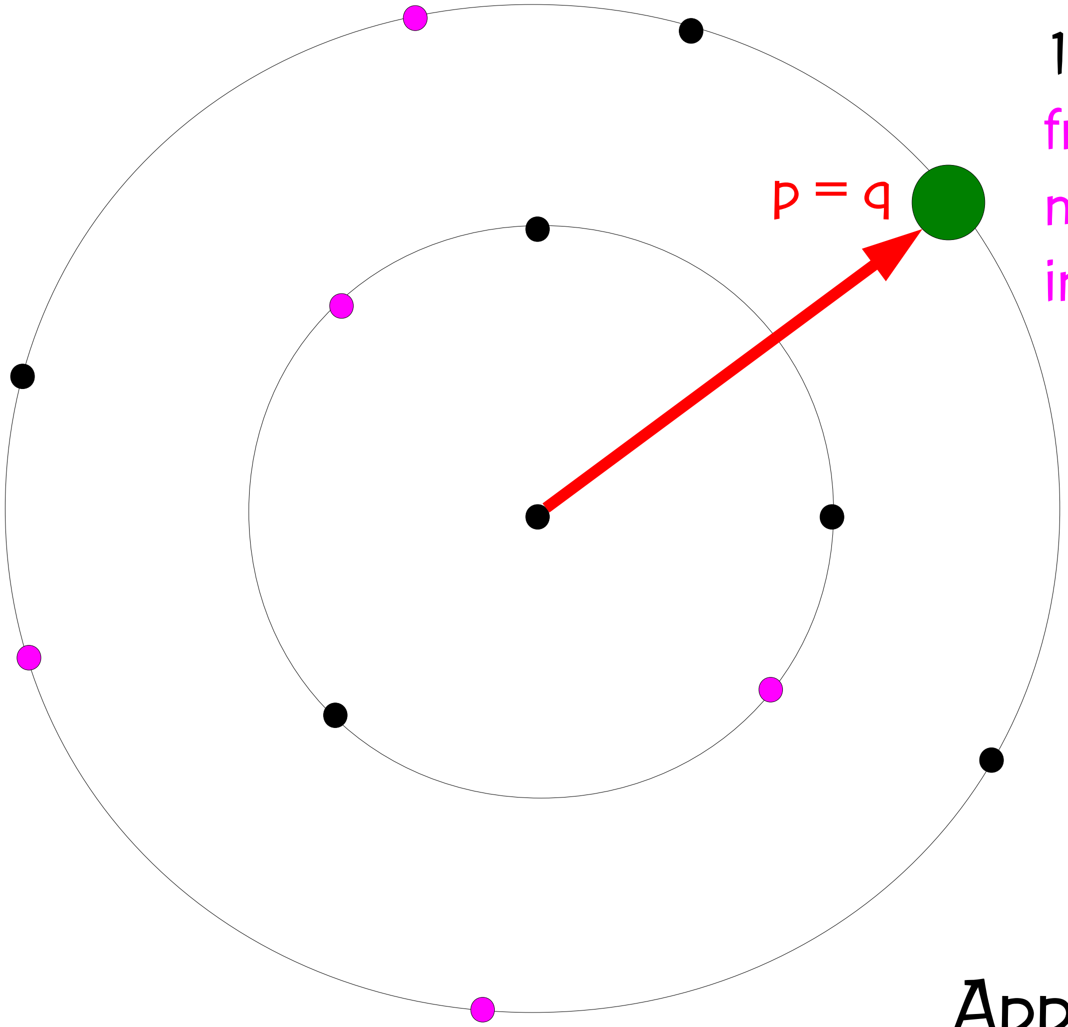
Approximate Local Search



1. Sample k improving solutions from 1-move and 2-move neighborhood of p and place them in an elite set E.

2. Select the best solution **q** from elite set E.

Approximate Local Search



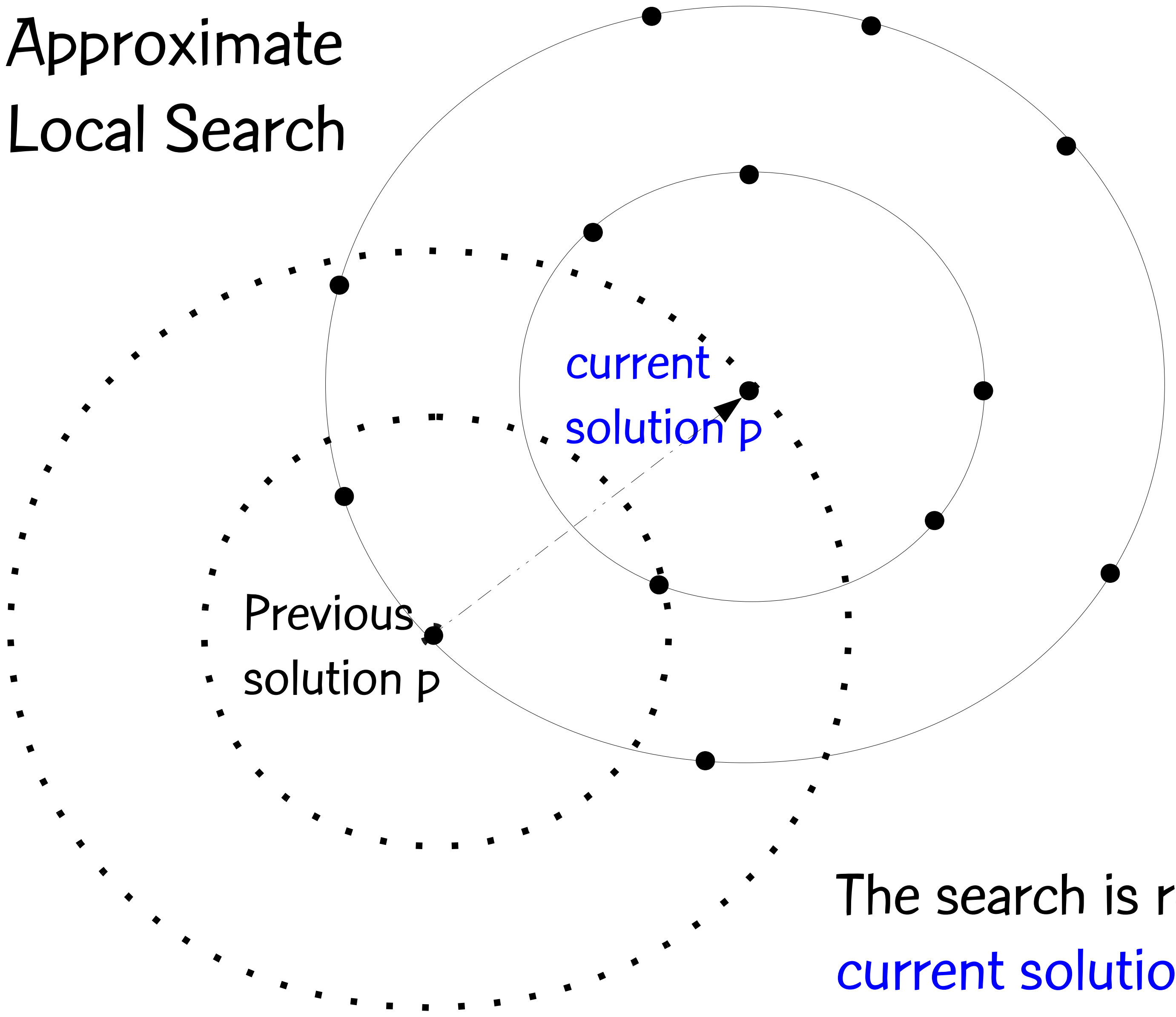
1. Sample k improving solutions from 1-move and 2-move neighborhood of p and place them in an elite set E.

2. Select the best solution **q** from elite set E.

3. Update p = q

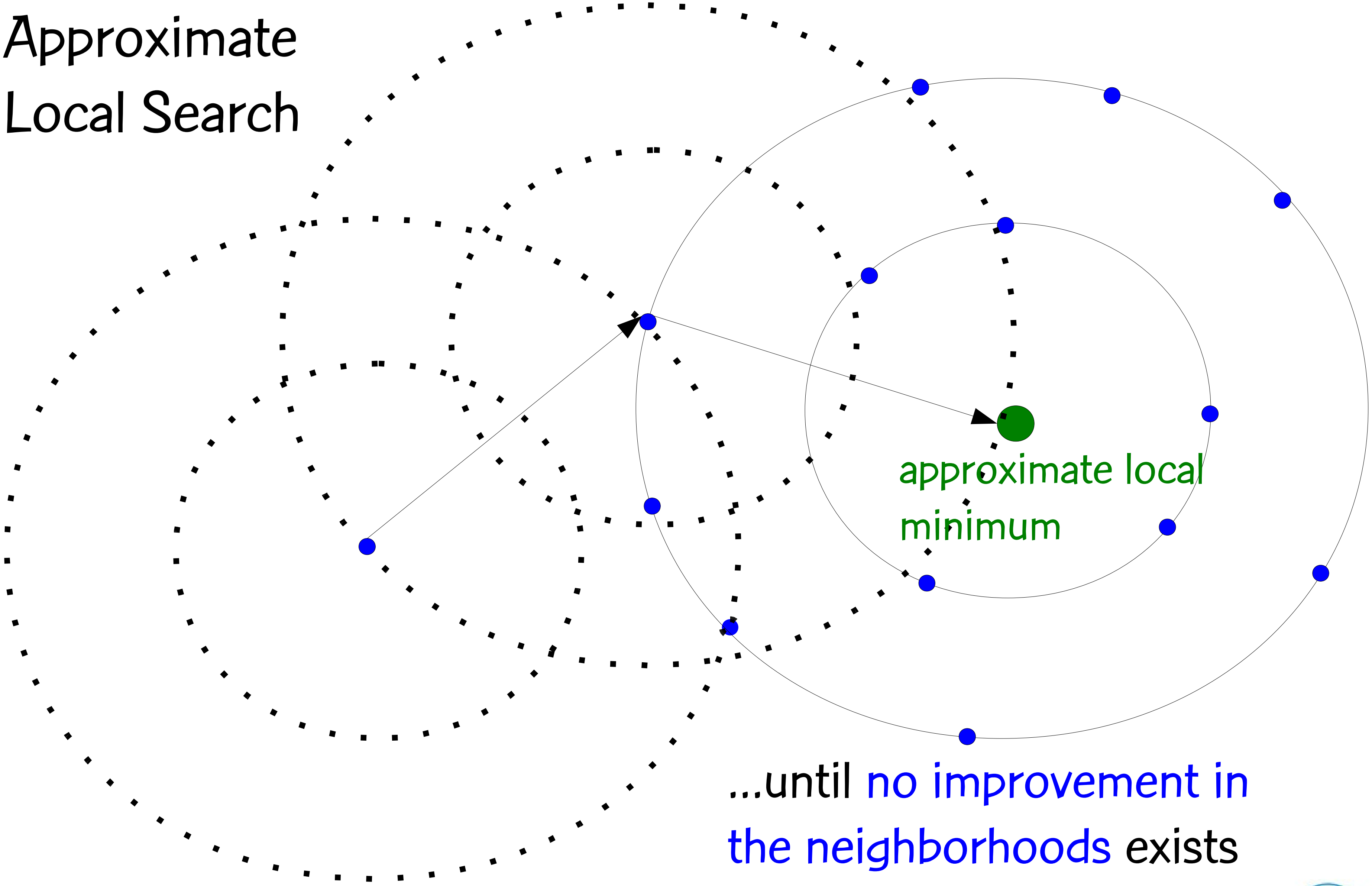
Approximate Local Search

Approximate Local Search



The search is repeated from current solution p until ….

Approximate Local Search

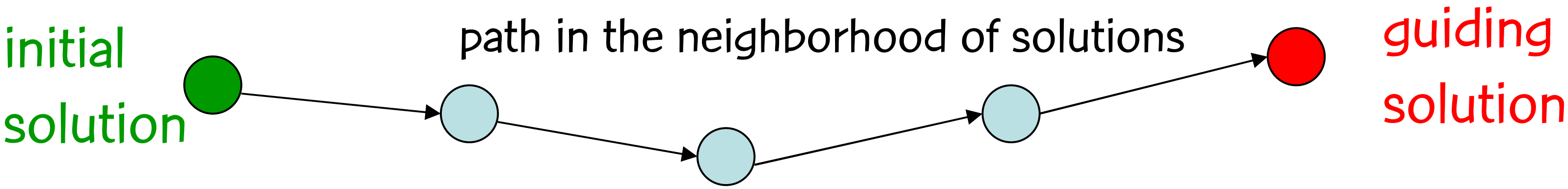


...until no improvement in the neighborhoods exists

Path-relinking

Path-relinking (Glover, 1996)

Exploration of trajectories that connect high quality (elite) solutions:



Path-relinking

Path is generated by selecting moves that introduce in the initial solution attributes of the guiding solution.

At each step, all moves that incorporate attributes of the guiding solution are evaluated and the best move is selected:

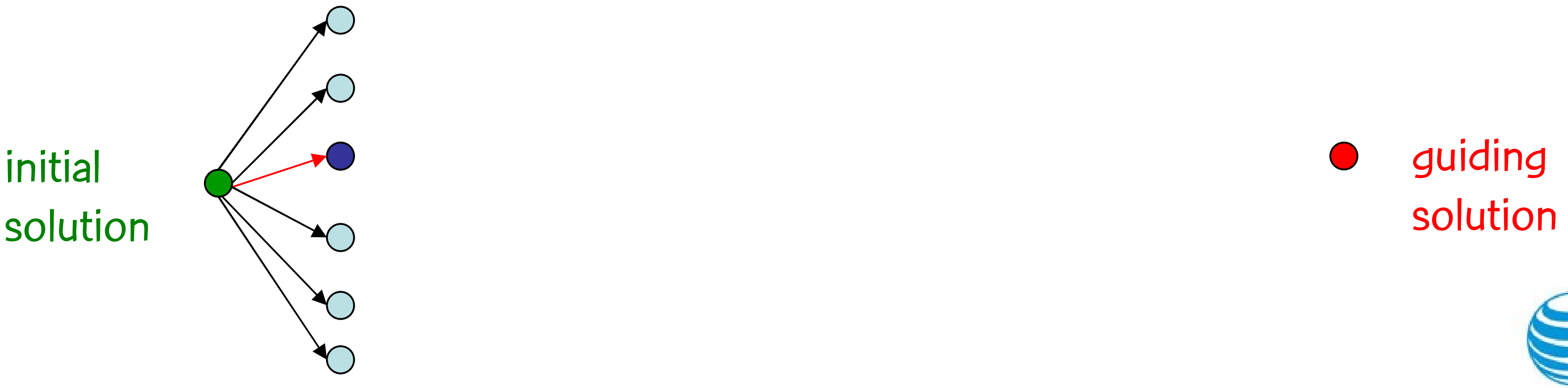
initial solution

guiding solution

Path-relinking

Path is generated by selecting moves that introduce in the initial solution attributes of the guiding solution.

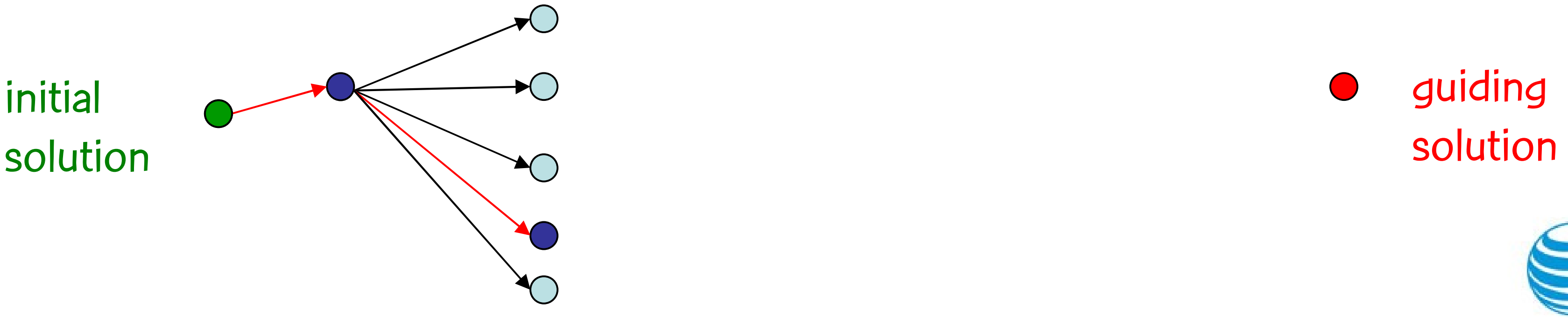
At each step, all moves that incorporate attributes of the guiding solution are evaluated and the best move is selected:



Path-relinking

Path is generated by selecting moves that introduce in the initial solution attributes of the guiding solution.

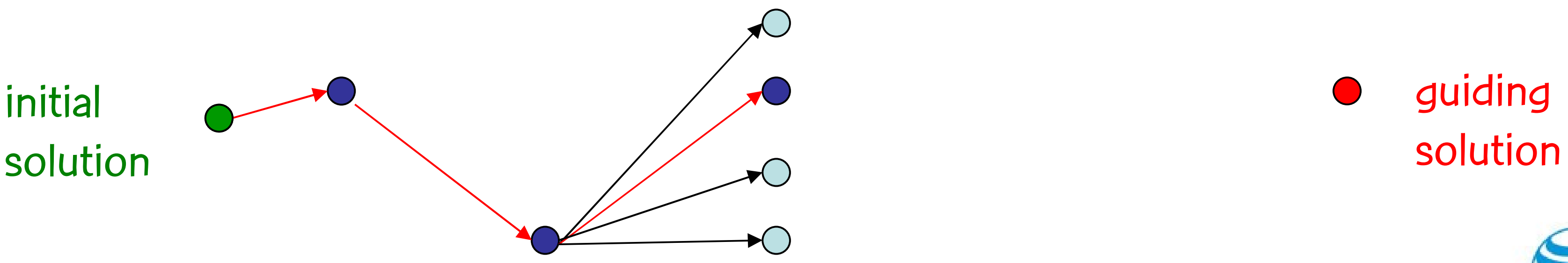
At each step, all moves that incorporate attributes of the guiding solution are evaluated and the best move is selected:



Path-relinking

Path is generated by selecting moves that introduce in the initial solution attributes of the guiding solution.

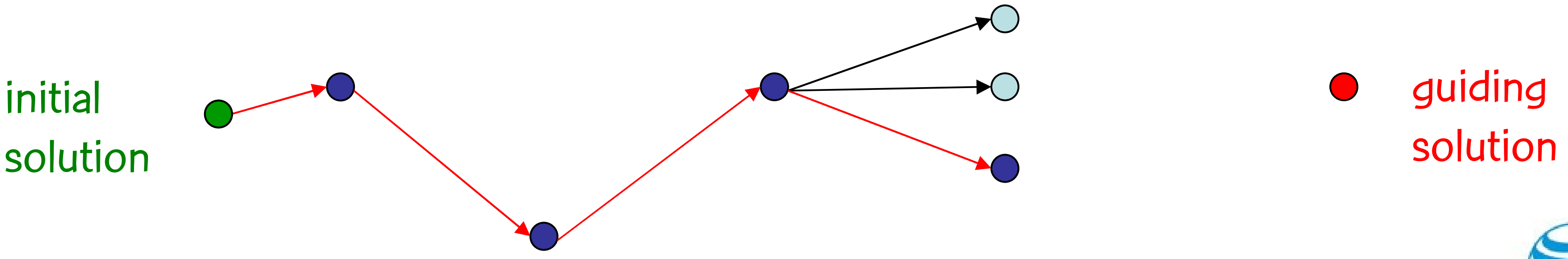
At each step, all moves that incorporate attributes of the guiding solution are evaluated and the best move is selected:



Path-relinking

Path is generated by selecting moves that introduce in the initial solution attributes of the guiding solution.

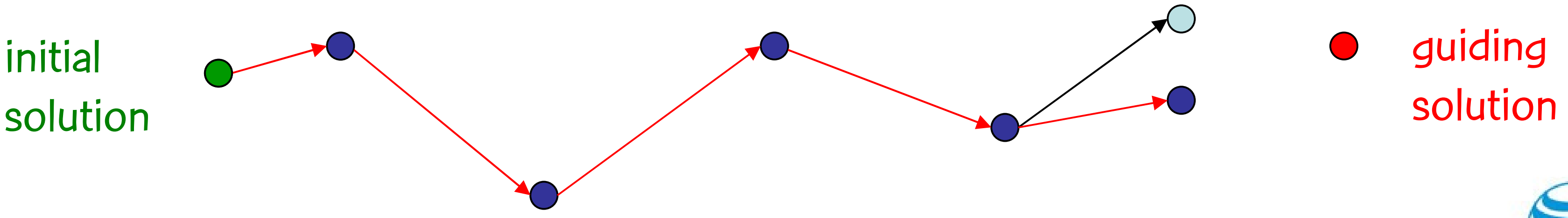
At each step, all moves that incorporate attributes of the guiding solution are evaluated and the best move is selected:



Path-relinking

Path is generated by selecting moves that introduce in the initial solution attributes of the guiding solution.

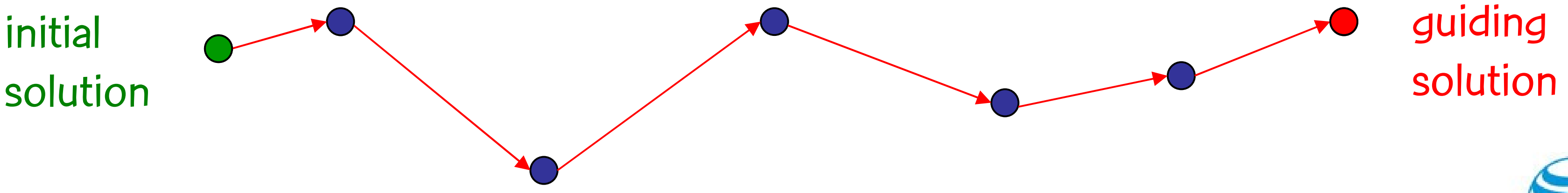
At each step, all moves that incorporate attributes of the guiding solution are evaluated and the best move is selected:



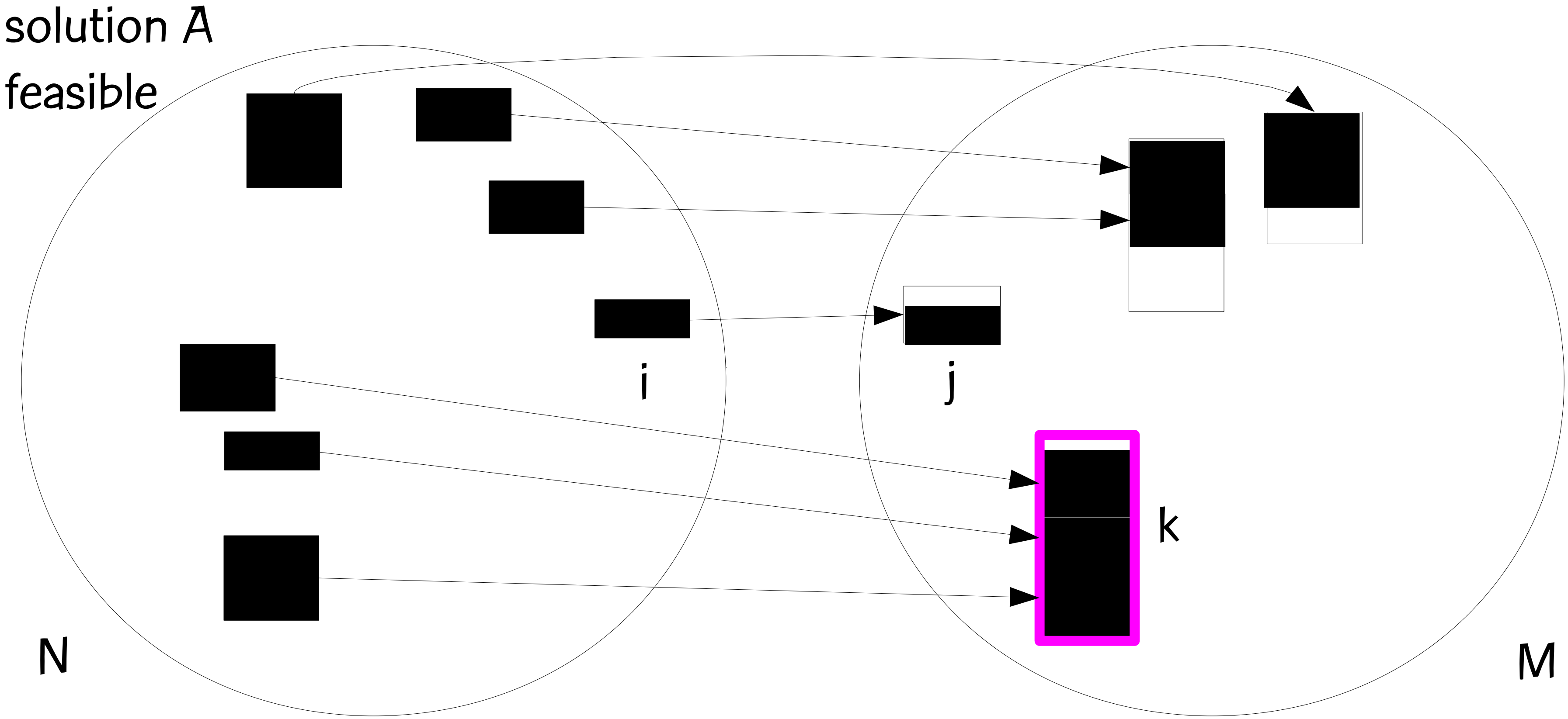
Path-relinking

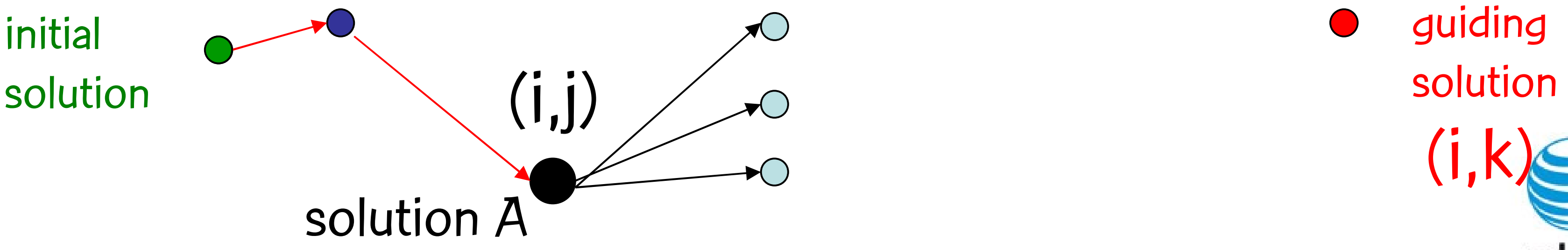
Path is generated by selecting moves that introduce in the initial solution attributes of the guiding solution.

At each step, all moves that incorporate attributes of the guiding solution are evaluated and the best move is selected:



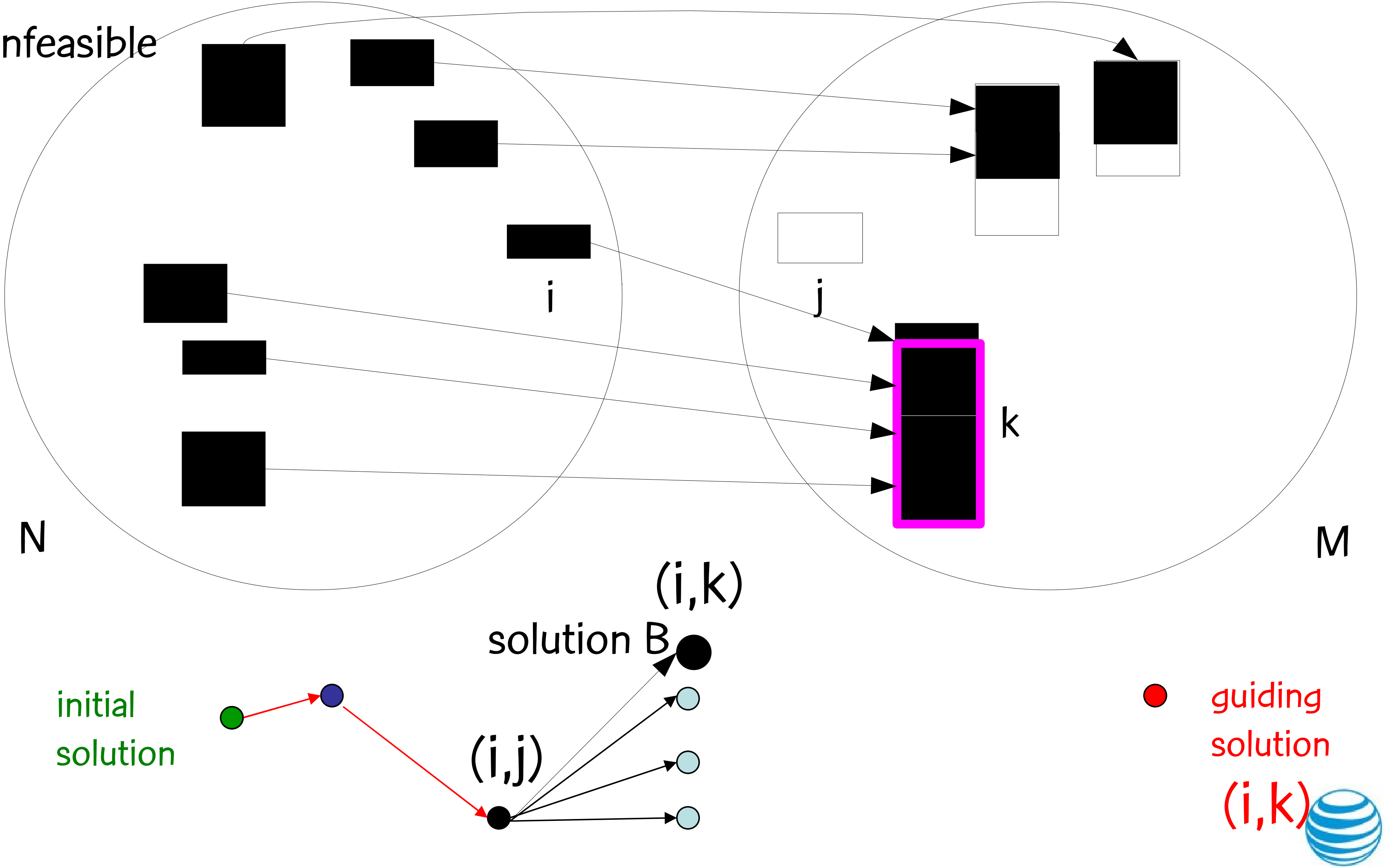
Infeasibility in path-relinking for GQAP



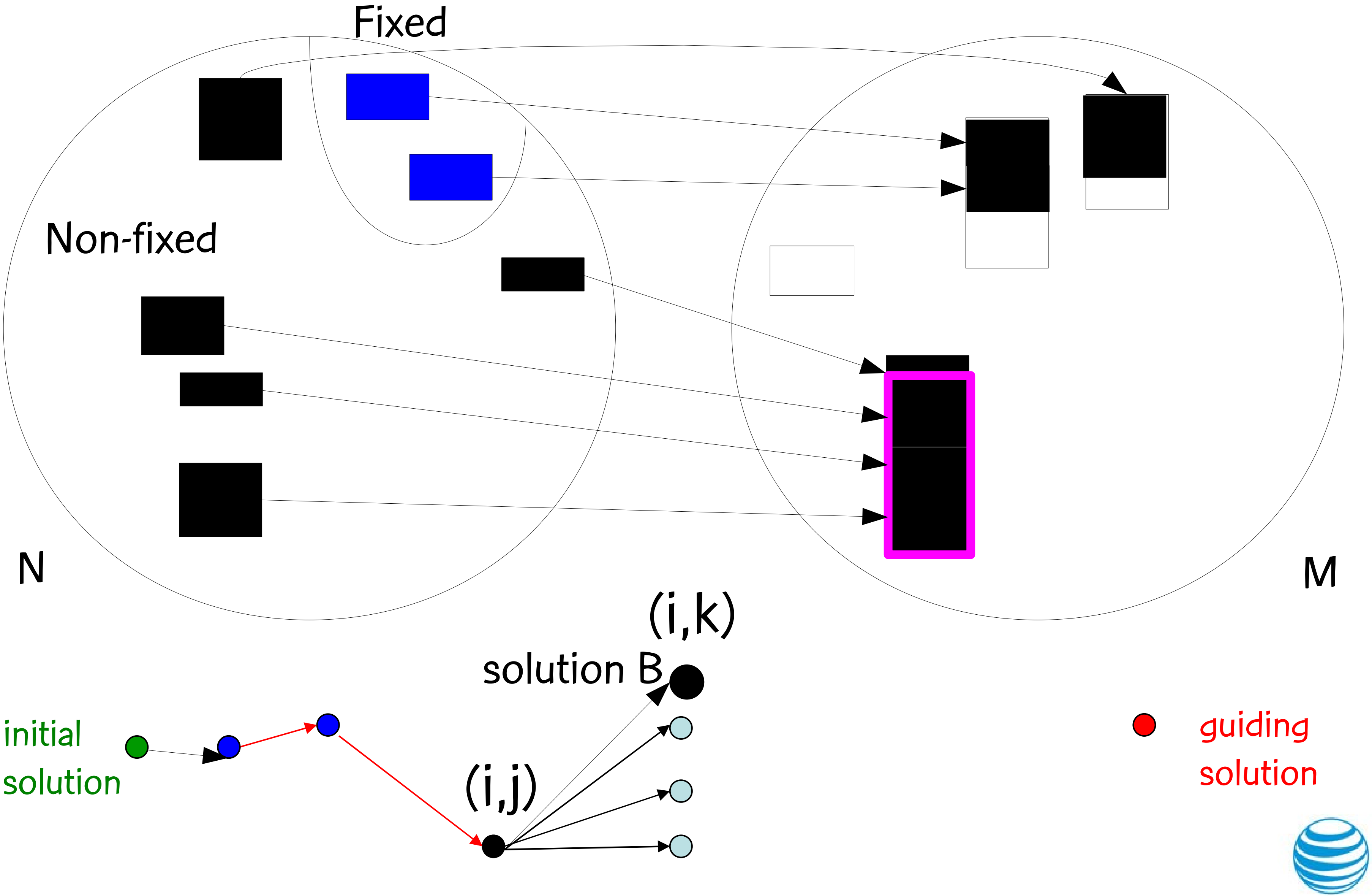


Infeasibility in path relinking for GQAP

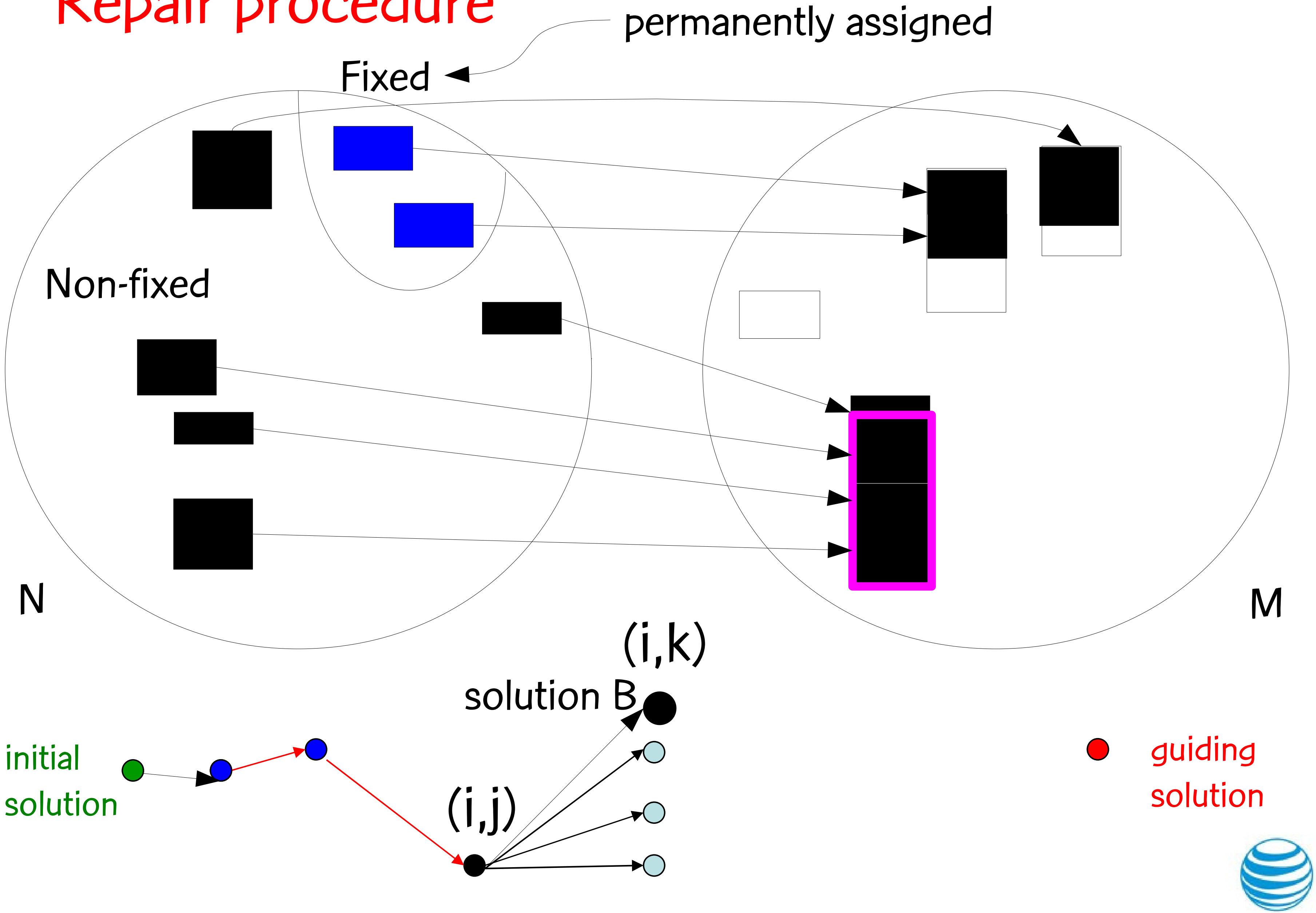
solution B
infeasible



Repair procedure

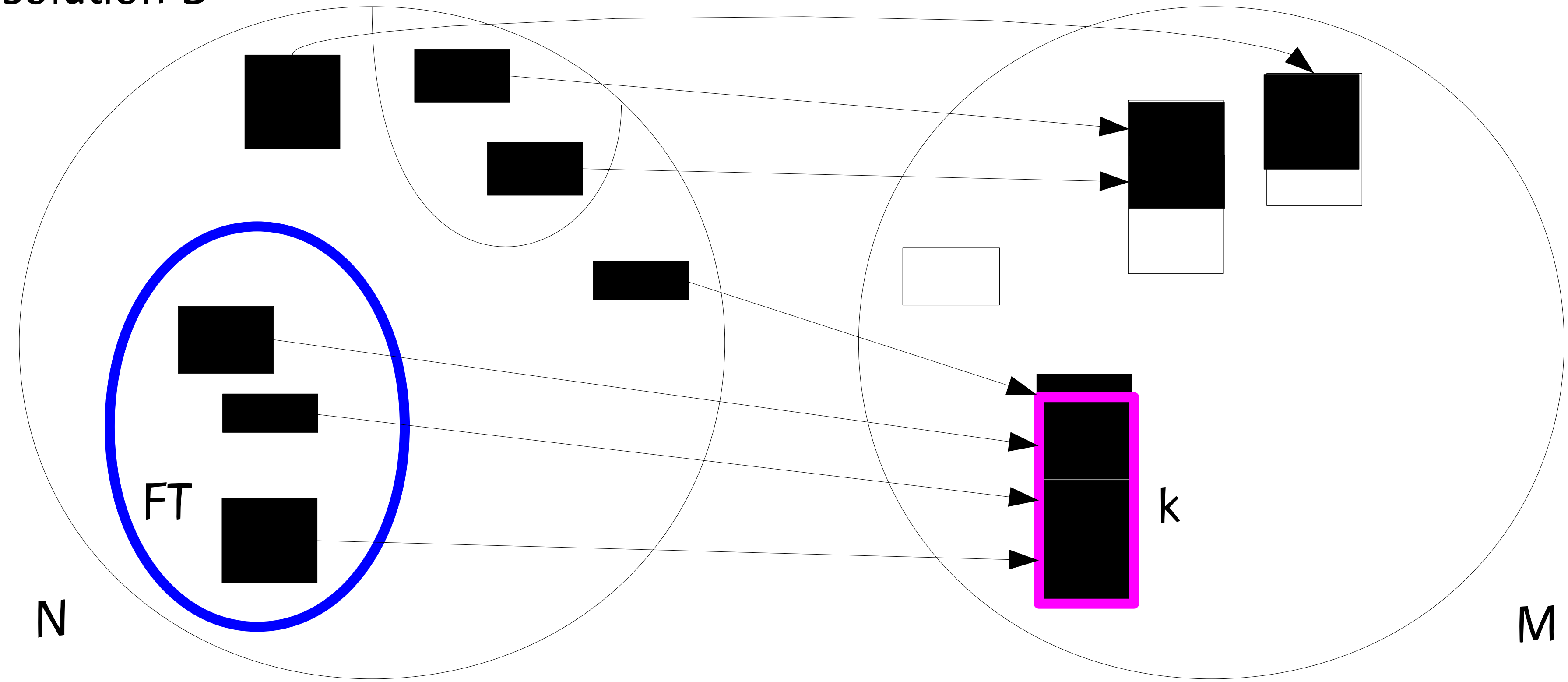


Repair procedure



Repair procedure

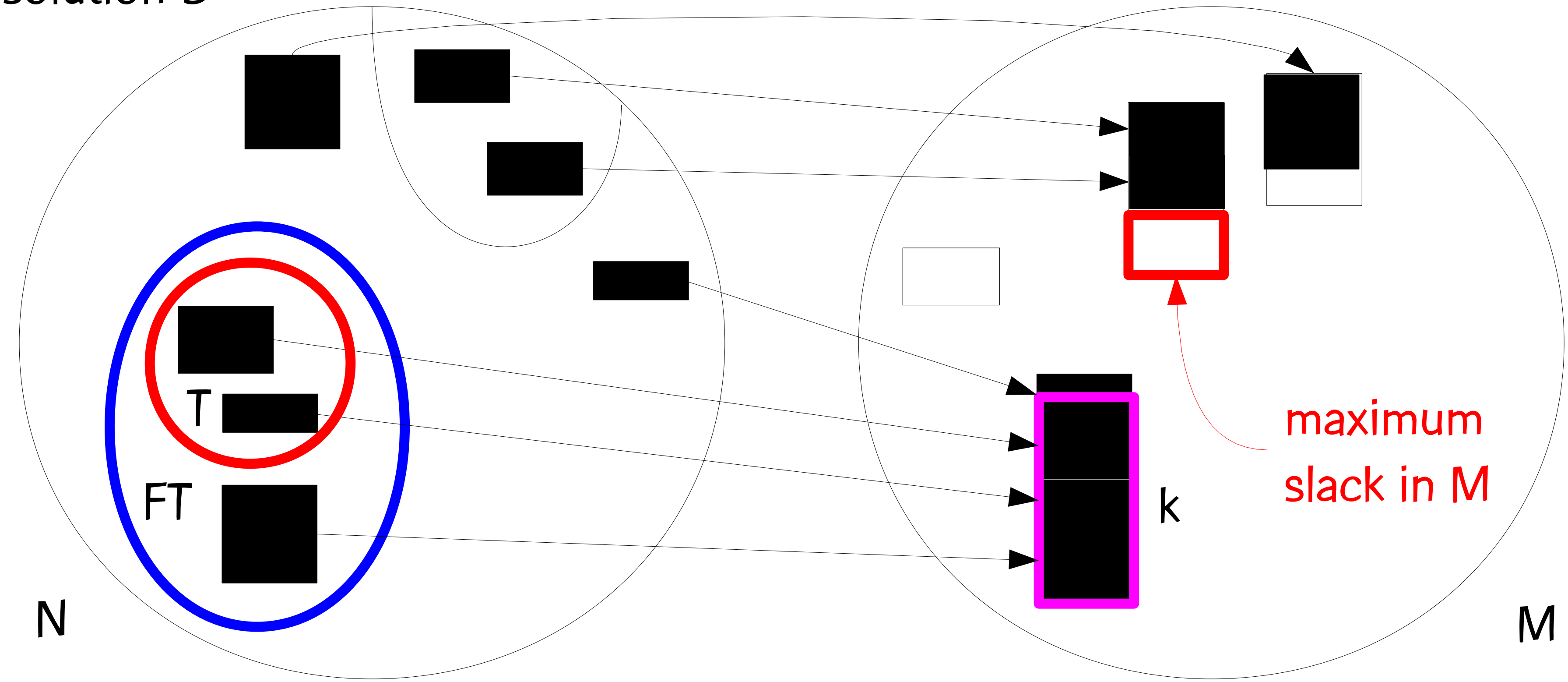
solution B



1. Set FT ⊆ non-Fixed: all facilities in solution B assigned to location k

Repair procedure

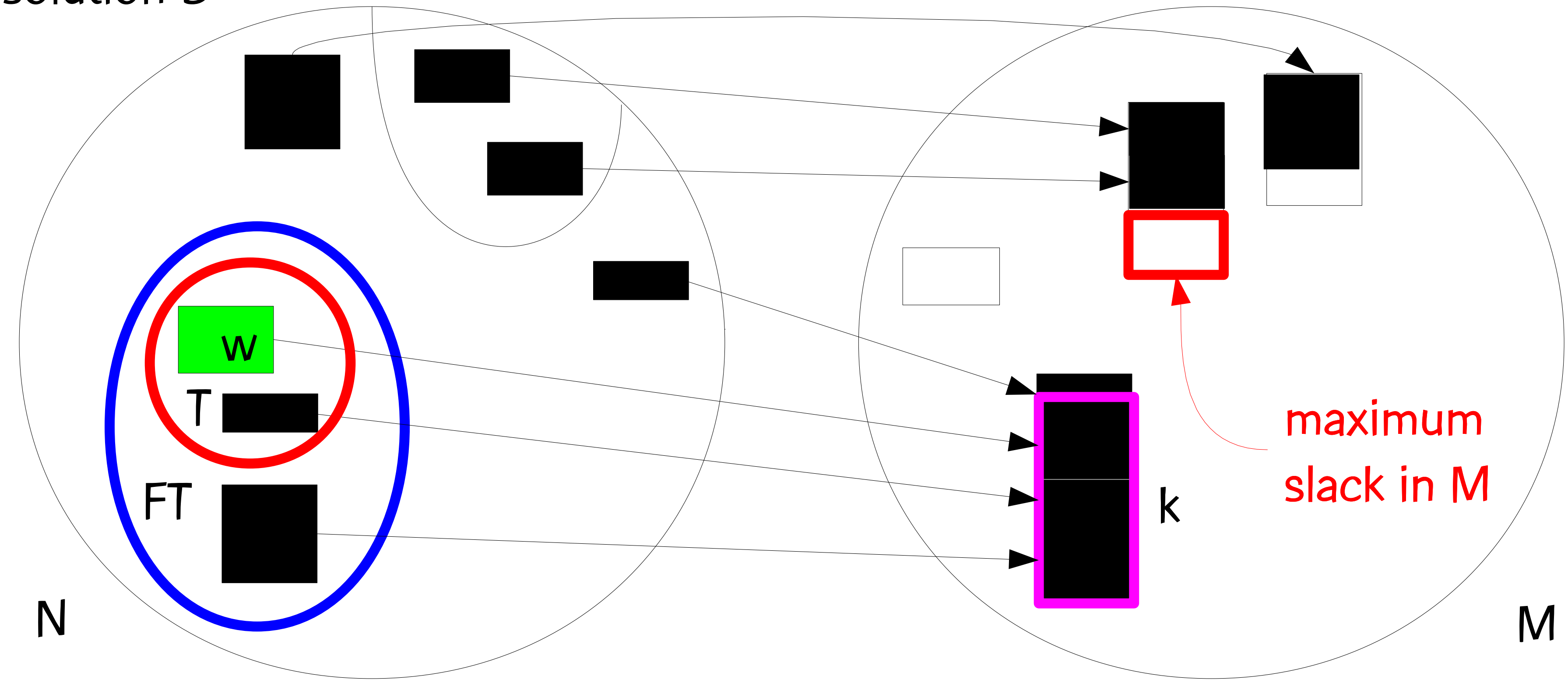
solution B



1. Set FT $\subseteq$ non-Fixed: all facilities in solution B assigned to location k
2. Set T $\subseteq$ FT: all facilities in B with demand $\leq$ maximum slack in M

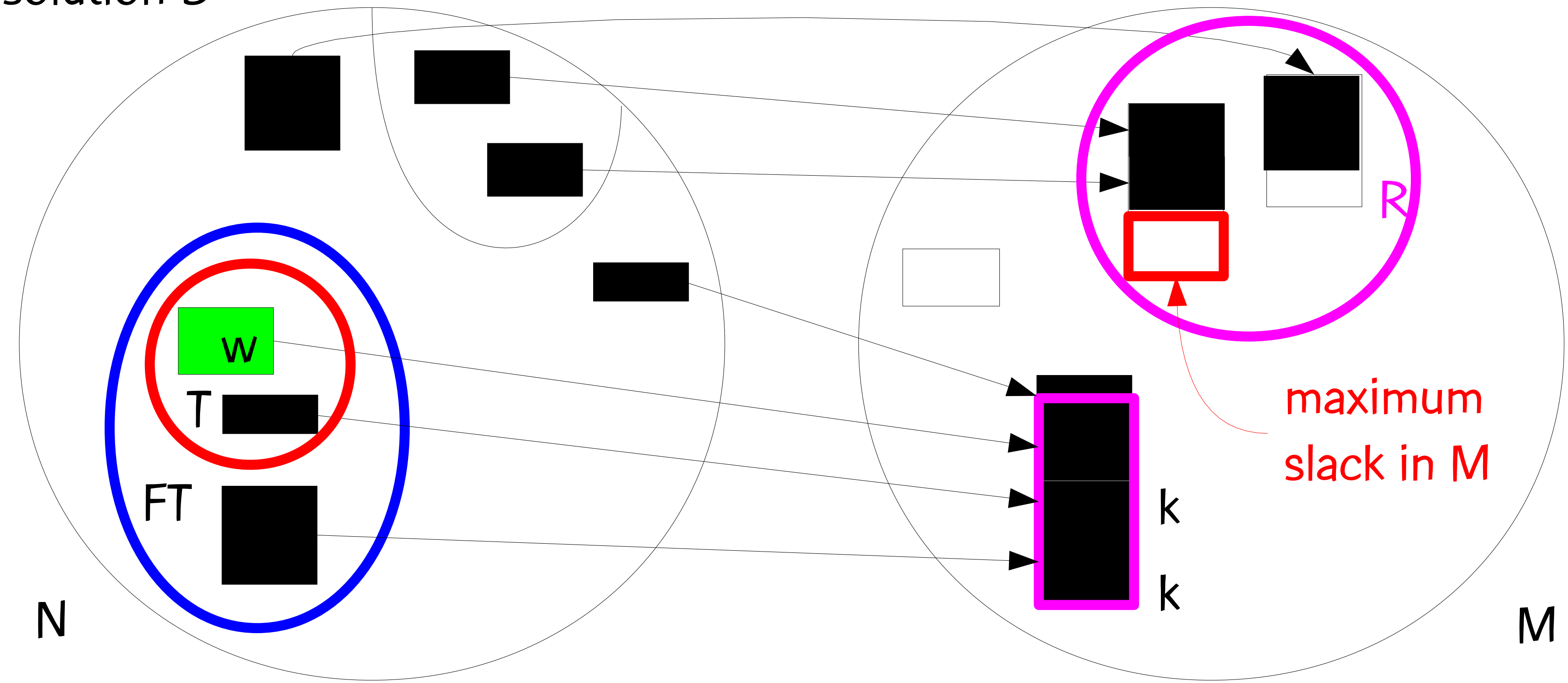
Repair procedure

solution B



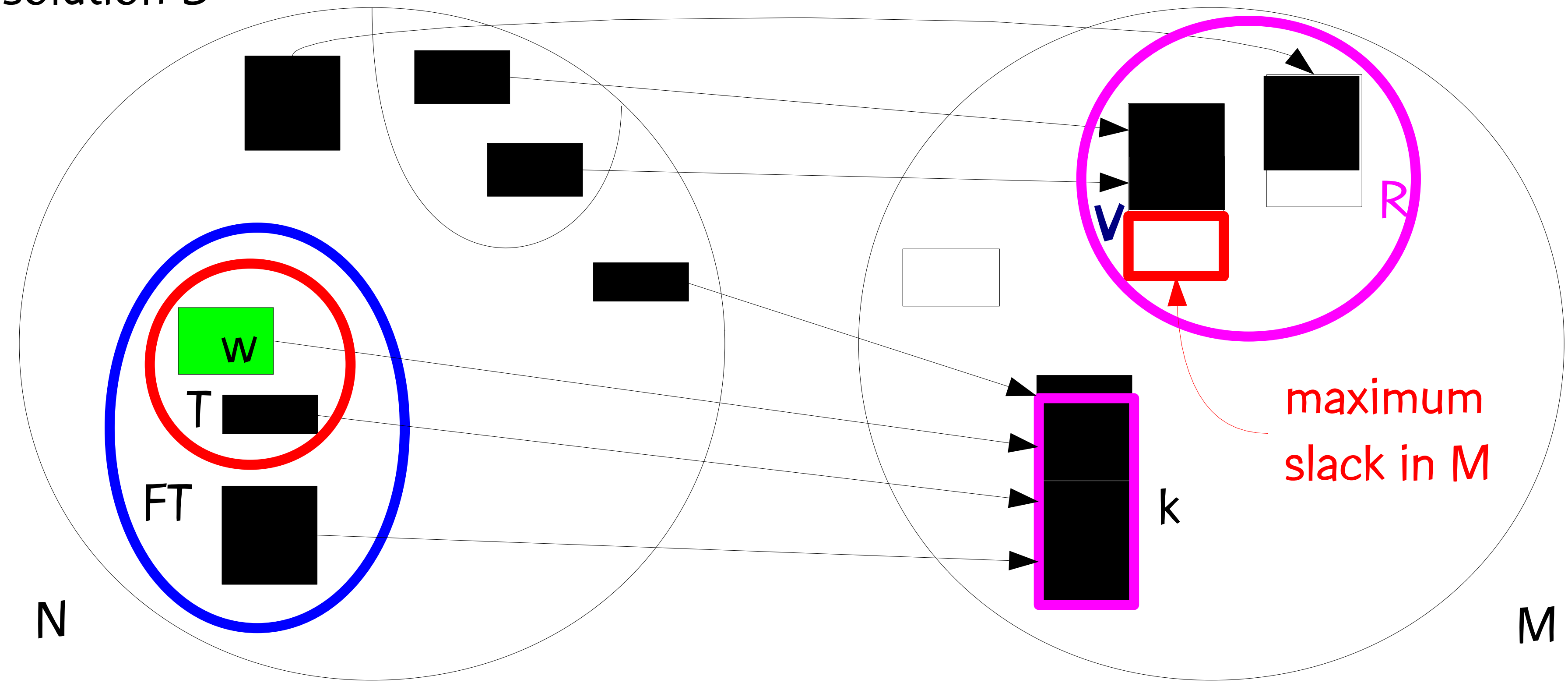
1. Set FT ⊆ non-Fixed: all facilities in solution B assigned to location k
2. Set T ⊆ FT: all facilities in B with demand ≤ maximum slack in M
3. Randomly select a facility w ∈ T favoring those with higher demand

solution B

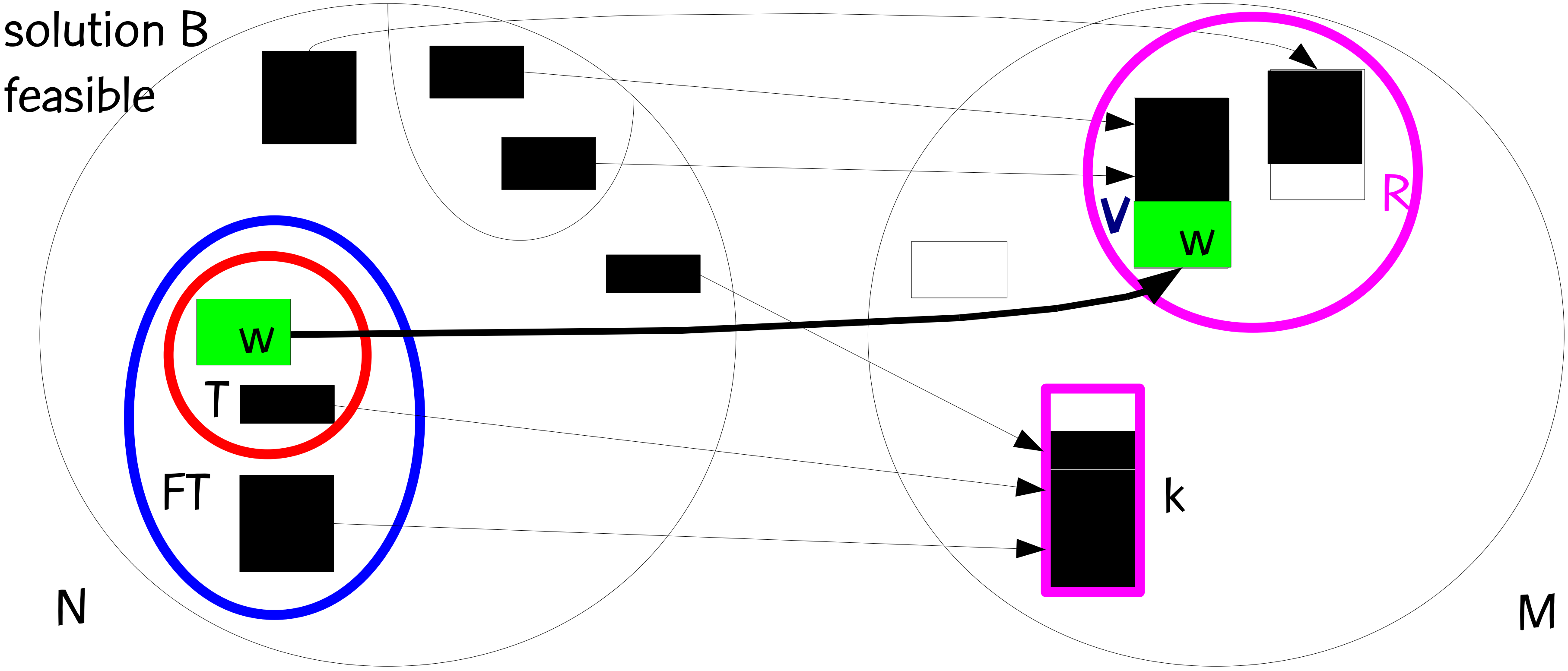


1. Set FT ⊆ non-Fixed: all facilities in solution B assigned to location k
2. Set T ⊆ FT: all facilities in B with demand ≤ maximum slack in M
3. Randomly select a facility w ∈ T favoring those with higher demand
4. Set R ⊆ M: all locations having slack ≥ demand of facility w

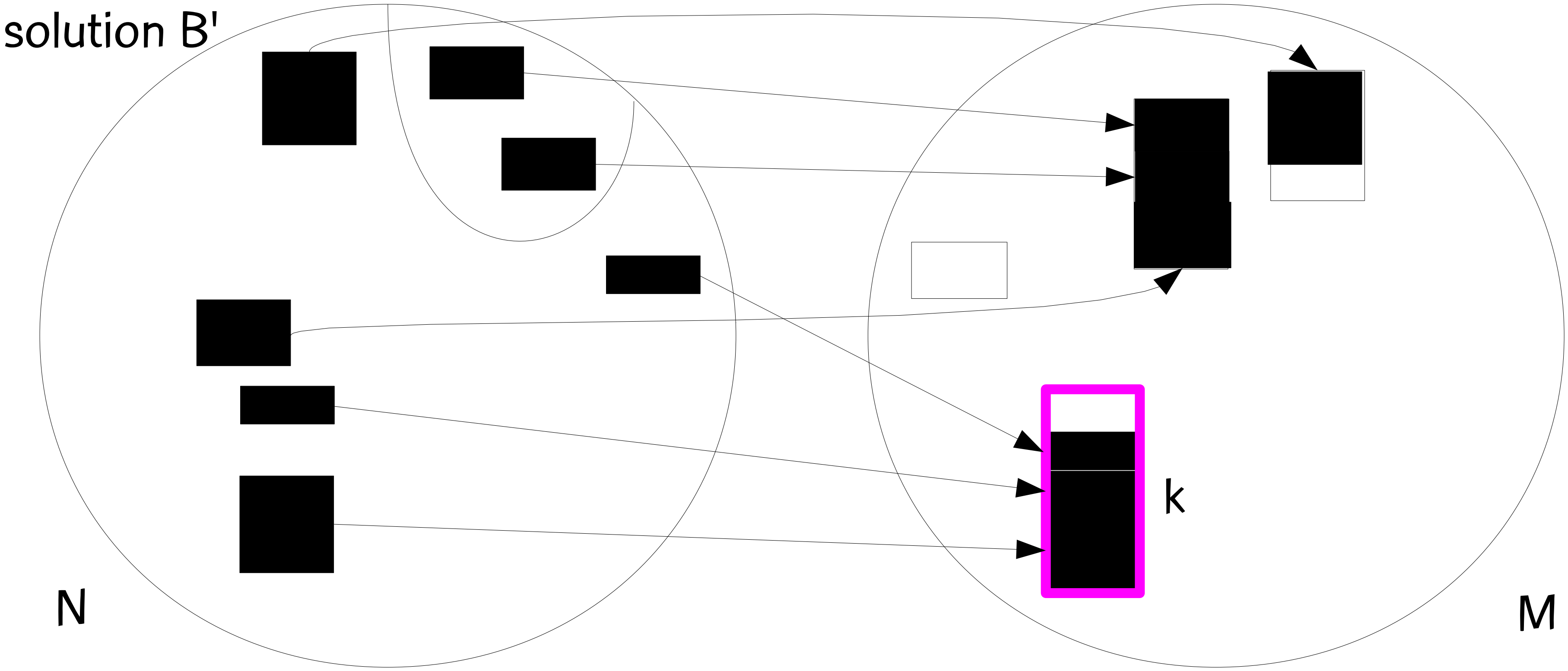
solution B

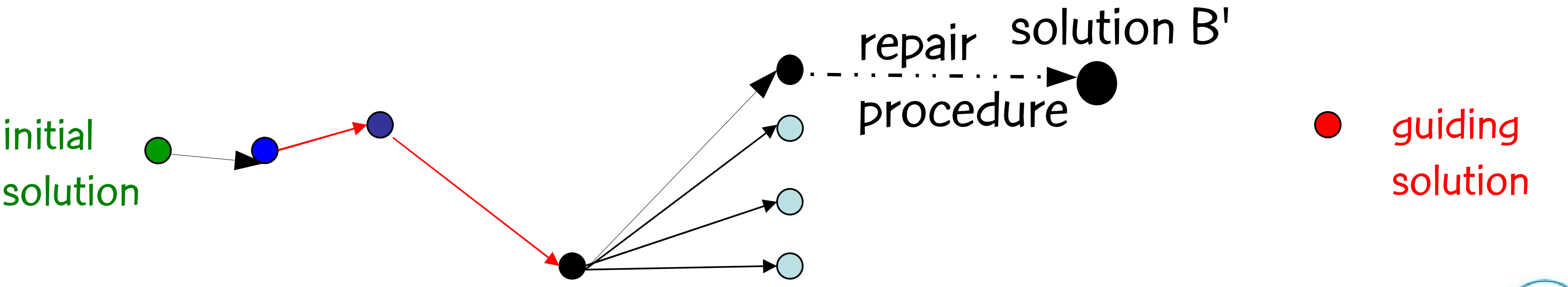


1. Set FT ⊆ non-Fixed: all facilities in solution B assigned to location k
2. Set T ⊆ FT: all facilities in B with demand ≤ maximum slack in M
3. Randomly select a facility w ∈ T favoring those with higher demand
4. Set R ⊆ M: all locations having slack ≥ demand of facility w
5. Randomly select a location v ∈ R (equal probability)

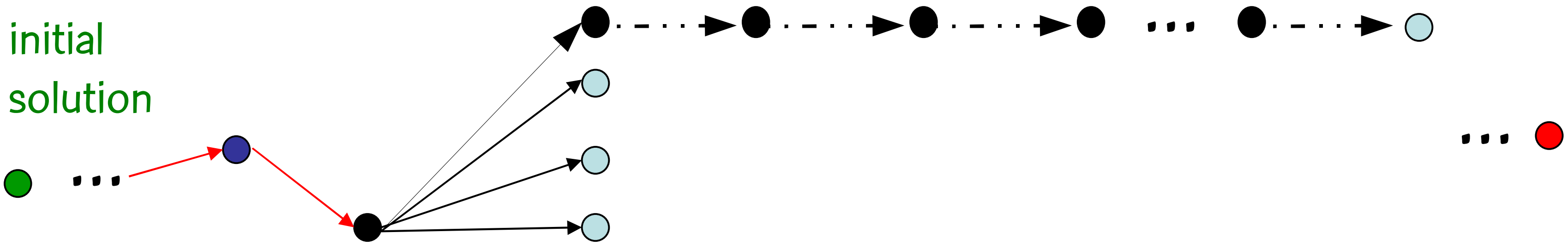


1. Set FT ⊆ non-Fixed: all facilities in solution B assigned to location k
2. Set T ⊆ FT: all facilities in B with demand ≤ maximum slack in M
3. Randomly select a facility w ∈ T favoring those with higher demand
4. Set R ⊆ M: all locations having slack ≥ demand of facility w
5. Randomly select a location v ∈ R (equal probability)
6. Assign facility w to location v


solution B'
k
N
M


repair
solution B'
procedure
initial
solution
guiding
solution

repair procedure

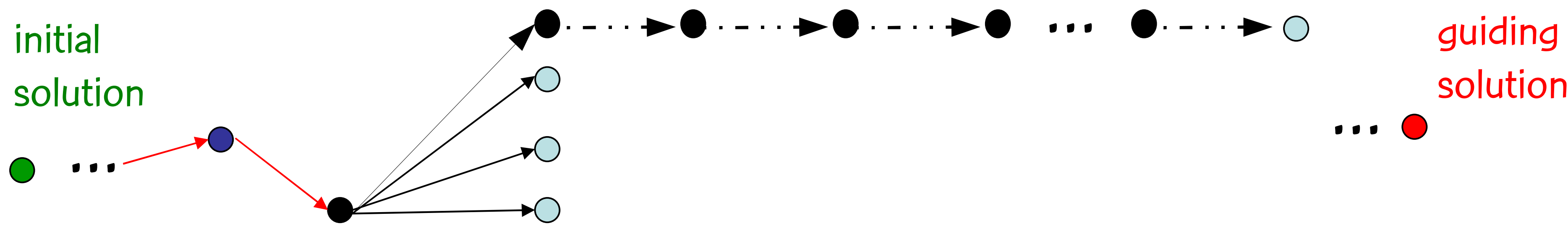


initial solution

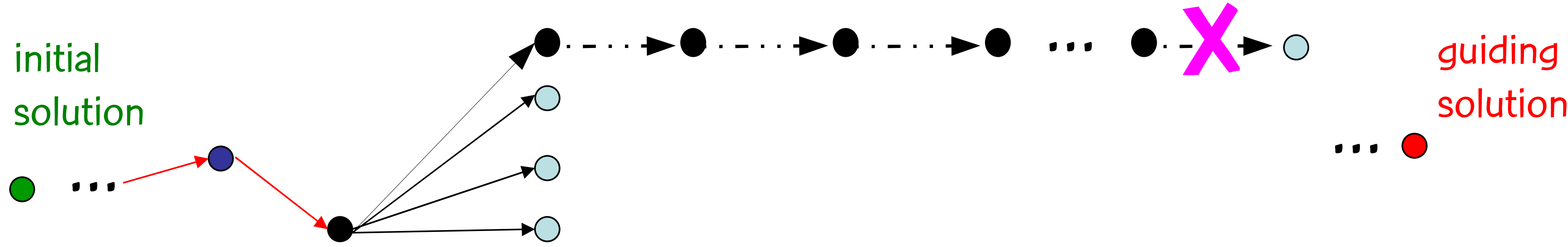
guiding solution

Possible outcomes

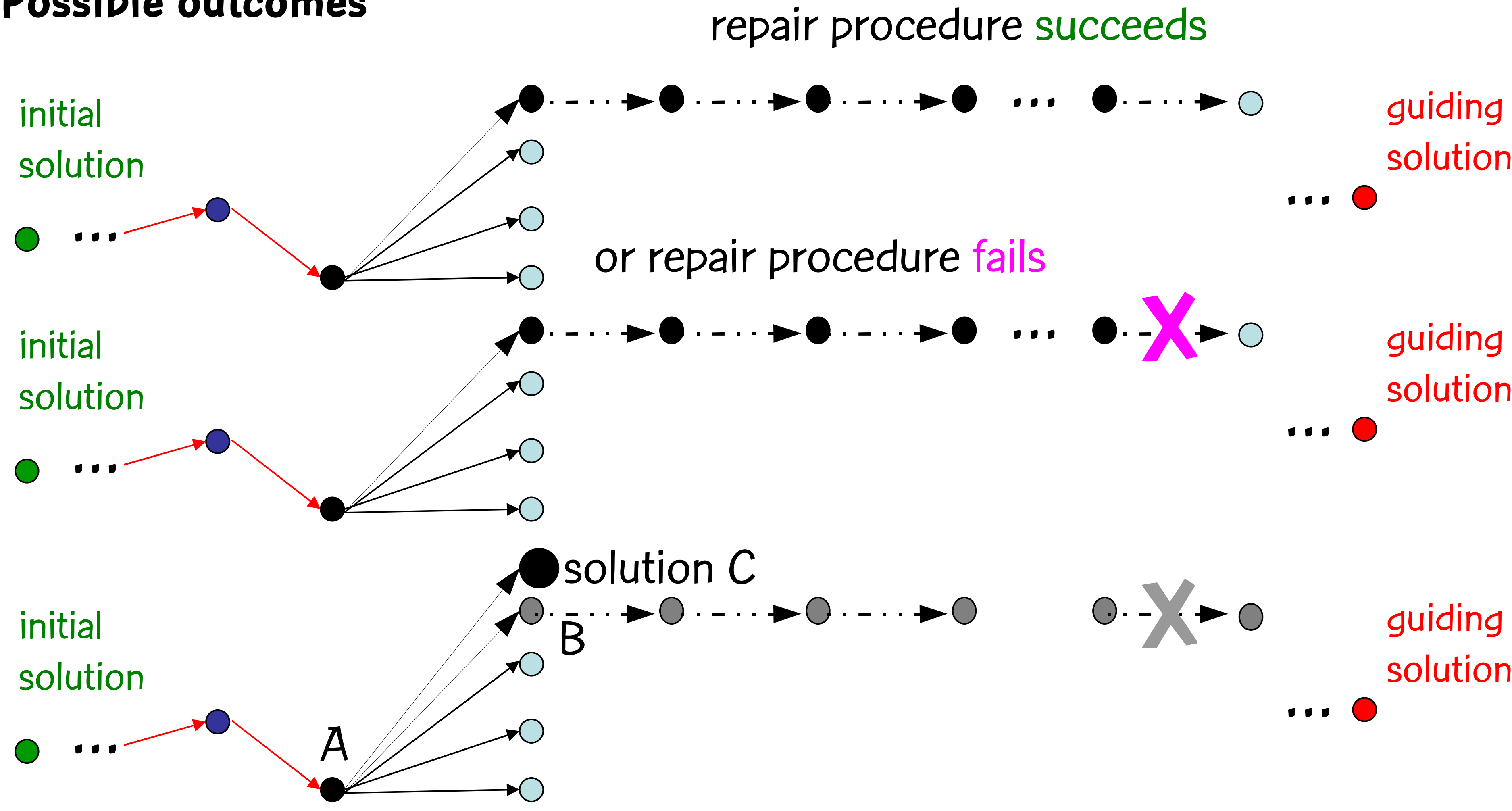
repair procedure succeeds



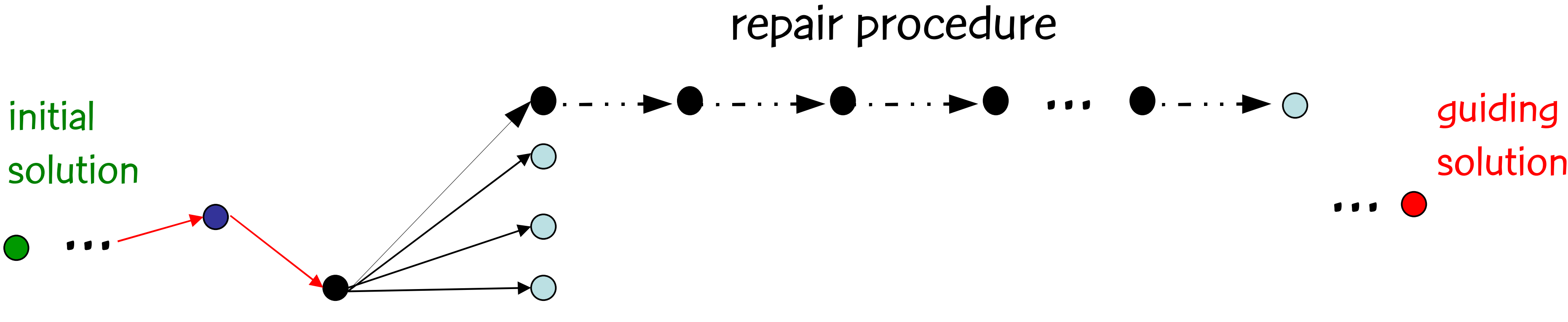
or repair procedure fails



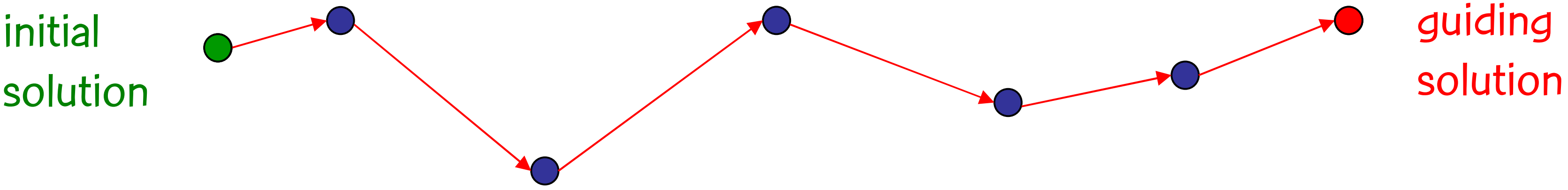
Possible outcomes

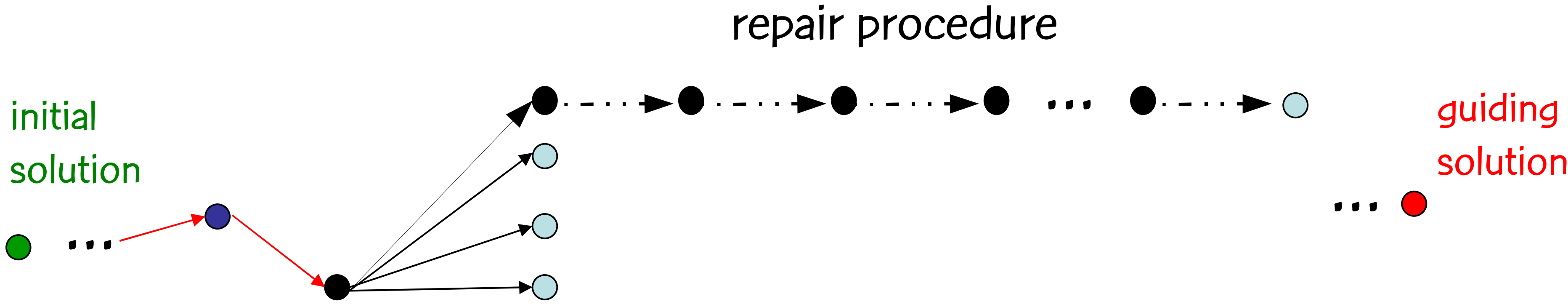


Repeat the repair procedure on solution B a maximum number of times. If a feasible solution is not found, discard B and move to solution C



So, instead of a path with feasible solution in one single step ...



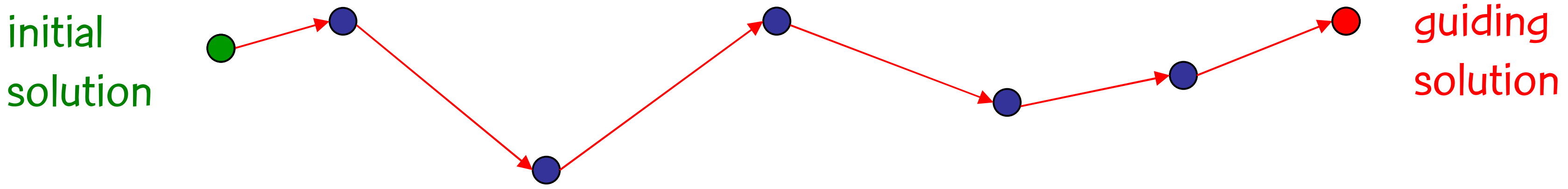


repair procedure

initial solution

guiding solution

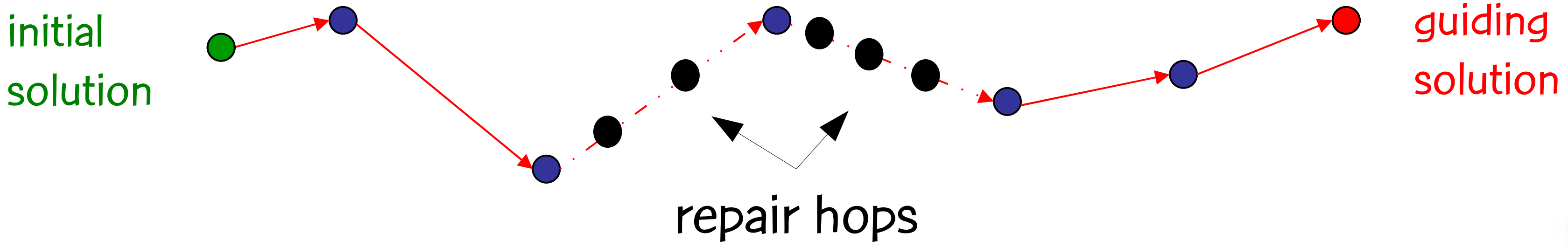
So, instead of a path with feasible solution in one single step ...



initial solution

guiding solution

We have now a path with eventual intermediate repair hops



initial solution

guiding solution

repair hops

Experimental results

Test environment

Dell PE1950 computer with a dual quad core 2.66 GHz Intel Xeon processors an 16 GB of Memory

Red Hat Linux version 5.1.19.6

Java language, Javac compiler ver.1.6.0-05

Random-number generator: Mersenne Twister algorithm (Matsumoto and Nishimura, 1998) from the COLT library

Test environment

Instances:

From Elloumi et al. (2003), Lee and Ma (2005), and Cordeau et al. (2006): 10 to 50 facilities and 3 to 20 locations.

Experimental Design:

For each instance we made 200 independent runs of GRASP-PR. Each run stopped when a solution value as good as the best in the literature was found.

Statistics:

Minimum, maximum, average times, and standard deviation.

Time for 95% of the runs to find solutions as good as the literature.

Parameter tuning for GRASP-PR

Instance: 50-10-95 (Cordeau et al., 2006).

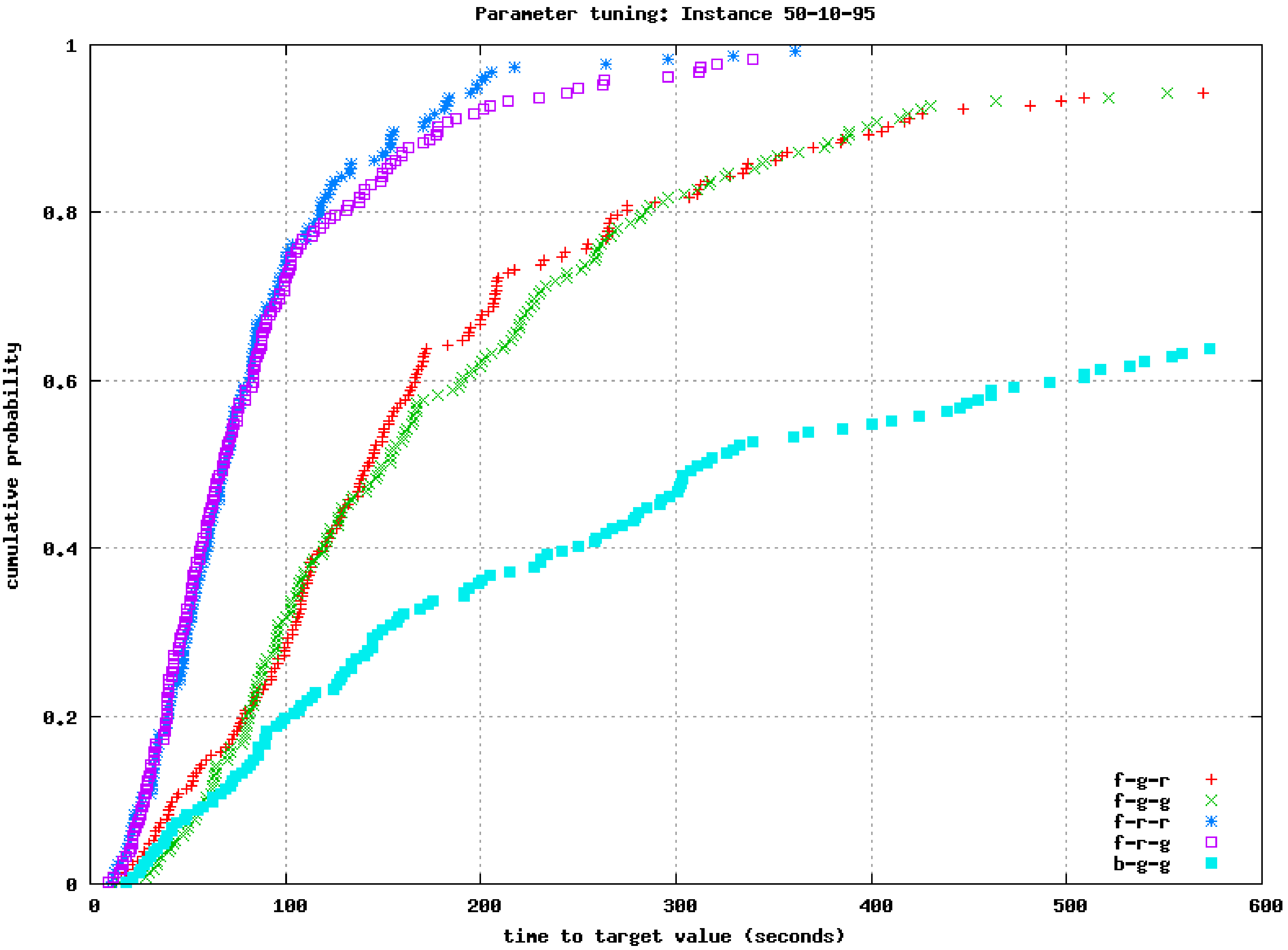
Strategies tested:

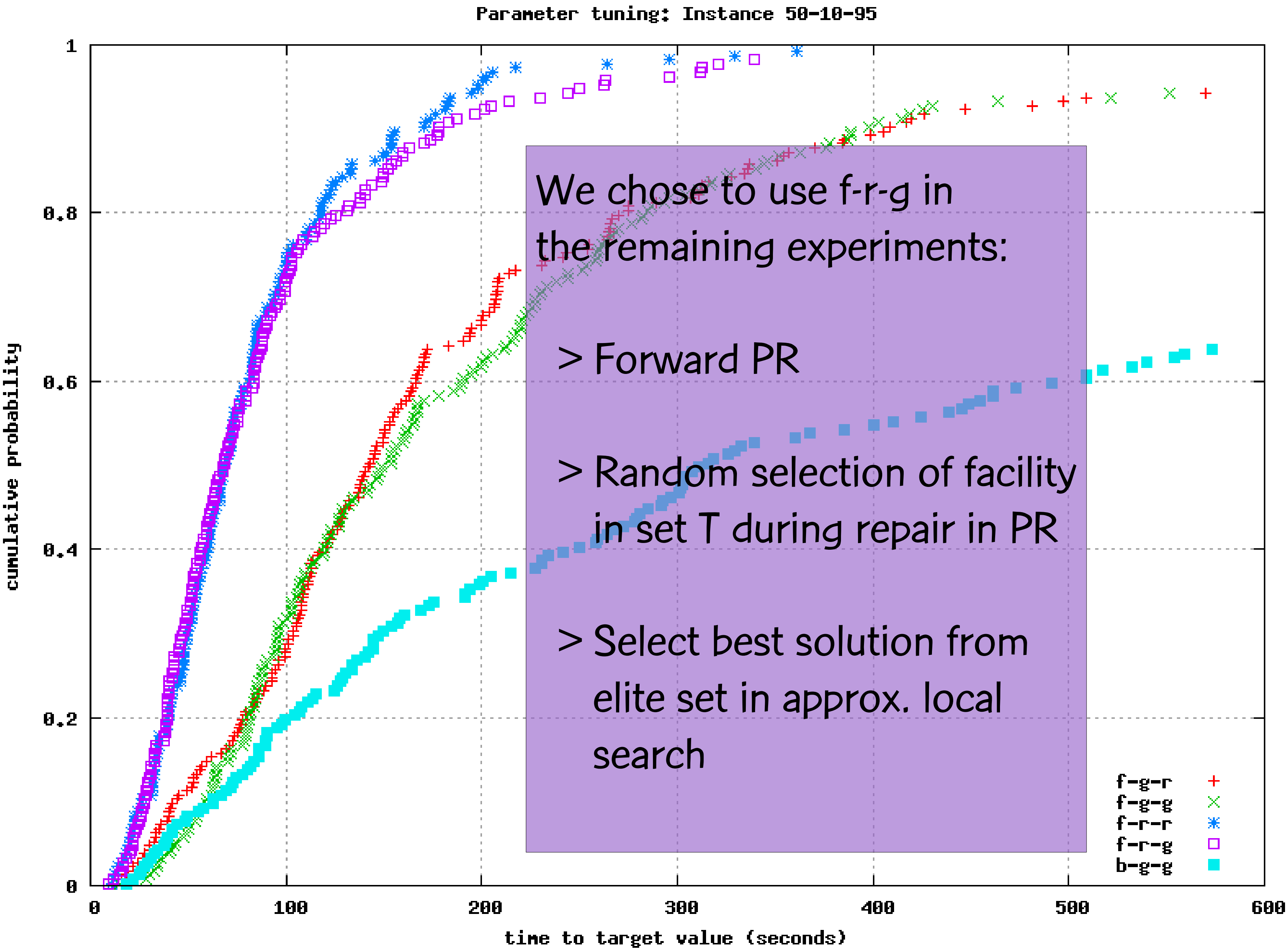
Path-relinking direction: forward (f) or backward (b);

Criteria to select a facility from set T in the repairing procedure: randomly (r) or greedily (g)

Criteria to select a solution from elite set in the approximate local search: randomly (r) or greedily (g).

Combinations: $2^3 = 8$


Parameter tuning: Instance 50-10-95
cumulative probability
0
0.2
0.4
0.6
0.8
1
0
100
200
300
400
500
600
time to target value (seconds)
f-g-r +
f-g-g ×
f-r-r ✱
f-r-g □
b-g-g ■



We chose to use f-r-g in the remaining experiments:

> Forward PR

> Random selection of facility in set T during repair in PR

> Select best solution from elite set in approx. local search

Comparison with other algorithms

Elloumi et al. (2003)

Lee and Ma (2005)

Cordieu et al. (2006)

Hahn et al. (2007)

Pessoa et al. (2008)

Comparison with Elloumi et al (2003):

Method(s): Three linearization methods (L1, L2, and L3), three semidefinite programming formulations (S0, S1, and S2) and a Lagrangian decomposition (D0).

Instances (Elloumi (1991) and Roupin (2004)): For each one of eight types [four configurations (A, B, C, and D) with two classes of instances], five instances with 10 facilities and three locations, and five instances with 20 facilities and five locations. Total of 80 instances

Comparison: GRASP-PR achieved the target values on all instances, with an AVERAGE performance improvement varying between a factor of 7.3 and over 5000 in relation to the BEST average time of the methods of Elloumi et al (2003)

Comparison with Lee and Ma (2005):

Method(s): Three linearization methods (F-Y, K-B, and L3), based on the work of Frieze and Yadegar (1983), Kaufman and Broeckx (1978), and Padberg and Rijal (1996) and a branch and bound method (B&B) based on the work of Burkard (1991).

Instances: Suite of test problems with 10 to 16 facilities and 3 to 8 locations. Total of 25 instances.

Comparison: GRASP-PR found the target value on all 200 runs for each of the instances, with an AVERAGE performance improvement varying between a factor of 11.2 and 1004.6 in relation to the BEST average time of the methods of Lee and Ma (2005)

Comparison with Cordeau et al. (2006):

Method: memetic algorithm.

Instances: problems with 20 to 50 facilities and 6 to 20 locations. Total = 21 instances

Comparison: GRASP-PR found the target value on all 200 runs for each of the instances, with an AVERAGE performance improvement varying between a factor of 1.5 and 59.2 in relation to the BEST average time of the memetic algorithm, except for instances 30-20-95, 35-15-95, and 50-10-75.

However, for the last two instances the FASTEST GRASP-PR running times were FAR LESS than those of the memetic algorithm.

For instance 30-20-95, the GRASP-PR heuristic found the best solution found by the memetic algorithm but in 44 hours and 47 minutes.

Comparison with Hahn et al. (2007):

Method(s): Level-1 reformulation-linearization technique (RLT) dual ascent procedure in a branch-and-bound scheme.

Instances: Four instances from Elloumi et al. (2003), three instances from Lee and Ma (2005), and one instance from Cordeau et al. (2006). Total of eight instances.

Comparison: GRASP-PR found the target value on all 200 runs for each of the instances, with an AVERAGE performance improvement varying between a factor of 8.8 and over 69,000 w.r.t. the BEST average time of the method of Hahn et al. (2007).

Comparison with Pessoa et al. (2008):

Method: Combination of Hahn et al. (2007) dual ascent procedure with the general-purpose volume algorithm of Barahona and Anbil (2000).

Instances: Four instances from Elloumi et al. (2003), three instances from Lee and Ma (2005), and 12 instances from Cordeau et al. (2006). Total of 24 instances.

Comparison: GRASP-PR found the target value on all 200 runs for each of the instances, with an AVERAGE performance improvement varying between a factor of 132.7 and over 100,000 w.r.t. the BEST average time of the method of Pessoa et al. (2008), except for instance 30-20-95.

Concluding remarks

Concluding remarks

Reviewed hospital layout optimization via QAP

Introduced hospital layout optimization via generalized QAP

Described several heuristics that can be applied to solve this layout problem:

> Greedy

> Randomized greedy

> Local search

> Path-relinking

> GRASP

> GRASP with path-relinking

Coauthor

Ricardo M.A. Silva
Fed. U. of Lavras,
Brazil. Visiting scholar
at AT&T Research (2008-2010)

The End

Slides and full paper can be downloaded from http://mauricioresende.com