A biased random-key genetic algorithm for the unequal area facility layout problem

Mauricio G. C. Resende
AT&T Labs Research
Middletown, New Jersey

mgcr@research.att.com

Talk given at INFORMS Annual Meeting
San Francisco, California ✤ November 10, 2014

Summary

- Biased random-key genetic algorithms (BRKGA)
  - Evolutionary dynamics
  - Problem independent / problem dependent components
  - Application Programming Interface (API) for BRKGA
- The unequal area facility layout problem
  - Unconstrained variant
  - Constrained variant
- BRKGA for the unequal area facility layout problem
  - Encoding and decoding for unrestricted case
  - Experimental results
- Concluding remarks

Joint work with José F. Gonçalves (U. do Porto, Portugal)

J.F. Gonçalves and M.G.C.R., "A biased random-key genetic algorithm for the unequal area facility layout problem," Tech. Report, AT&T Labs Research, Middletown, NJ 07748 USA, August 2014.

Download from http://mauricio.resende.info

This research is partially funded by the European Regional Development Fund through the Programme COMPETE and by the Portuguese Government through FCT – Foundation for Science and Technology, project PTDC/EGE-GES/117692/2010.

at&t
Your world. Delivered.

Genetic algorithms and random keys

GAs and random keys

- Introduced by Bean (1994) for sequencing problems.

GAs and random keys

- Introduced by Bean (1994) for sequencing problems.
- Individuals are strings of real-valued numbers (random keys) in the interval [0,1).

$$S = (\ 0.25,\ 0.19,\ 0.67,\ 0.05,\ 0.89\ )$$

s(1)	s(2)	s(3)	s(4)	s(5)

GAs and random keys

- Introduced by Bean (1994) for sequencing problems.
- Individuals are strings of real-valued numbers (random keys) in the interval [0,1).
- Sorting random keys results in a sequencing order.

$S = (\ 0.25,\ 0.19,\ 0.67,\ 0.05,\ 0.89\ )$

$\quad\quad s(1) \quad s(2) \quad s(3) \quad s(4) \quad s(5)$

$S' = (\ 0.05,\ 0.19,\ 0.25,\ 0.67,\ 0.89\ )$

$\quad\quad s(4) \quad s(2) \quad s(1) \quad s(3) \quad s(5)$

Sequence: 4 – 2 – 1 – 3 – 5

GAs and random keys

- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)

$$a = (0.25, 0.19, 0.67, 0.05, 0.89)$$
$$b = (0.63, 0.90, 0.76, 0.93, 0.08)$$

GAs and random keys

- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passes the allele (key, or value of gene) to the child.

a = (0.25, 0.19, 0.67, 0.05, 0.89)
b = (0.63, 0.90, 0.76, 0.93, 0.08)

GAs and random keys

- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passes the allele (key, or value of gene) to the child.

a = (0.25, 0.19, 0.67, 0.05, 0.89)
b = (0.63, 0.90, 0.76, 0.93, 0.08)
c = ( )

GAs and random keys

- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passes the allele (key, or value of gene) to the child.

a = (0.25, 0.19, 0.67, 0.05, 0.89)
b = (0.63, 0.90, 0.76, 0.93, 0.08)
c = (0.25)

GAs and random keys

- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passes the allele (key, or value of gene) to the child.

a = (0.25, 0.19, 0.67, 0.05, 0.89)
b = (0.63, 0.90, 0.76, 0.93, 0.08)
c = (0.25, 0.90)

GAs and random keys

- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passes the allele (key, or value of gene) to the child.

a = (0.25, 0.19, 0.67, 0.05, 0.89)
b = (0.63, 0.90, 0.76, 0.93, 0.08)
c = (0.25, 0.90, 0.76)

GAs and random keys

- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passes the allele (key, or value of gene) to the child.

a = (0.25, 0.19, 0.67, 0.05, 0.89)
b = (0.63, 0.90, 0.76, 0.93, 0.08)
c = (0.25, 0.90, 0.76, 0.05)

GAs and random keys

- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passes the allele (key, or value of gene) to the child.

a = (0.25, 0.19, 0.67, 0.05, 0.89)
b = (0.63, 0.90, 0.76, 0.93, 0.08)
c = (0.25, 0.90, 0.76, 0.05, 0.89)

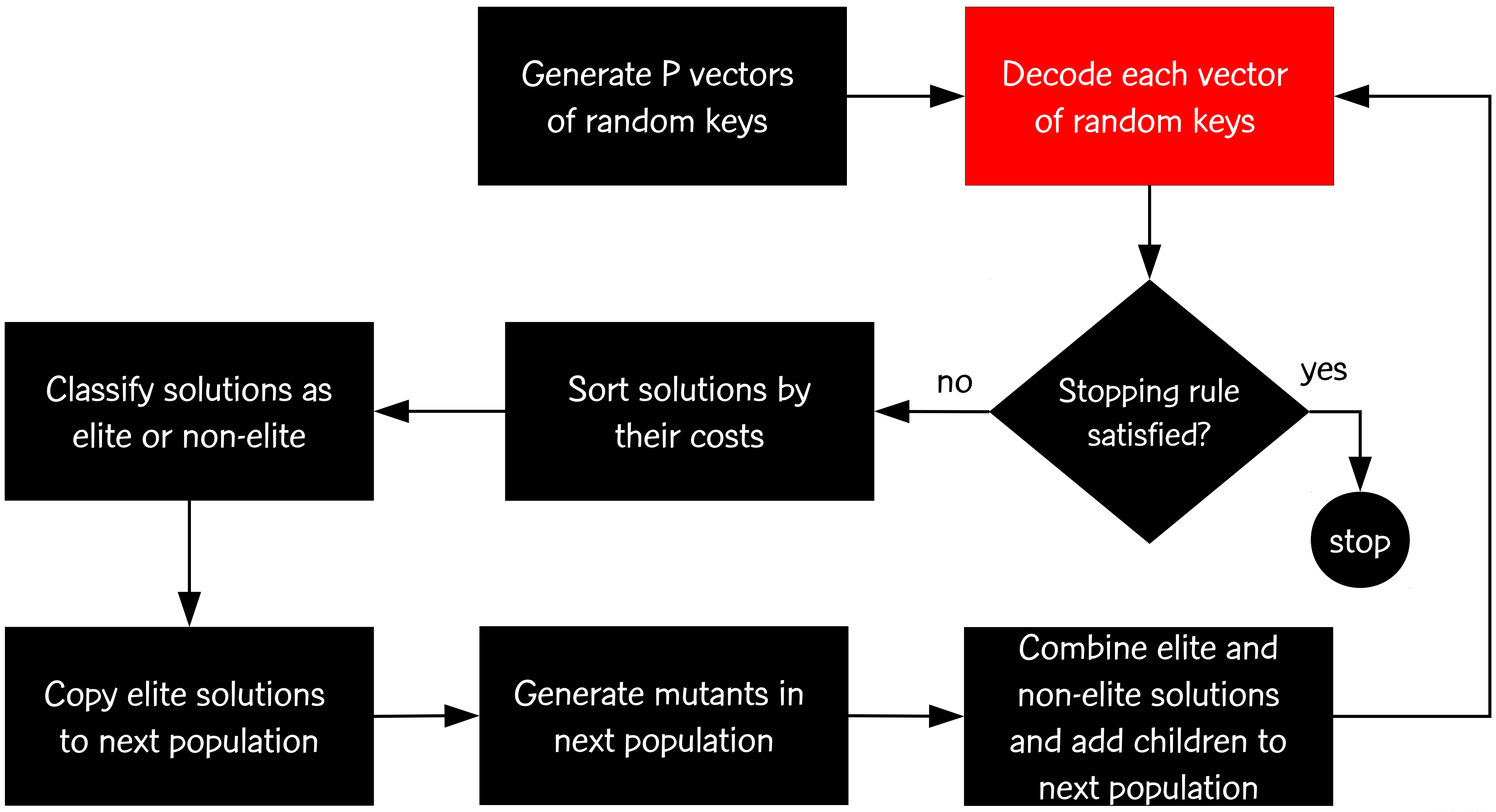
GAs and random keys

- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passes the allele (key, or value of gene) to the child.

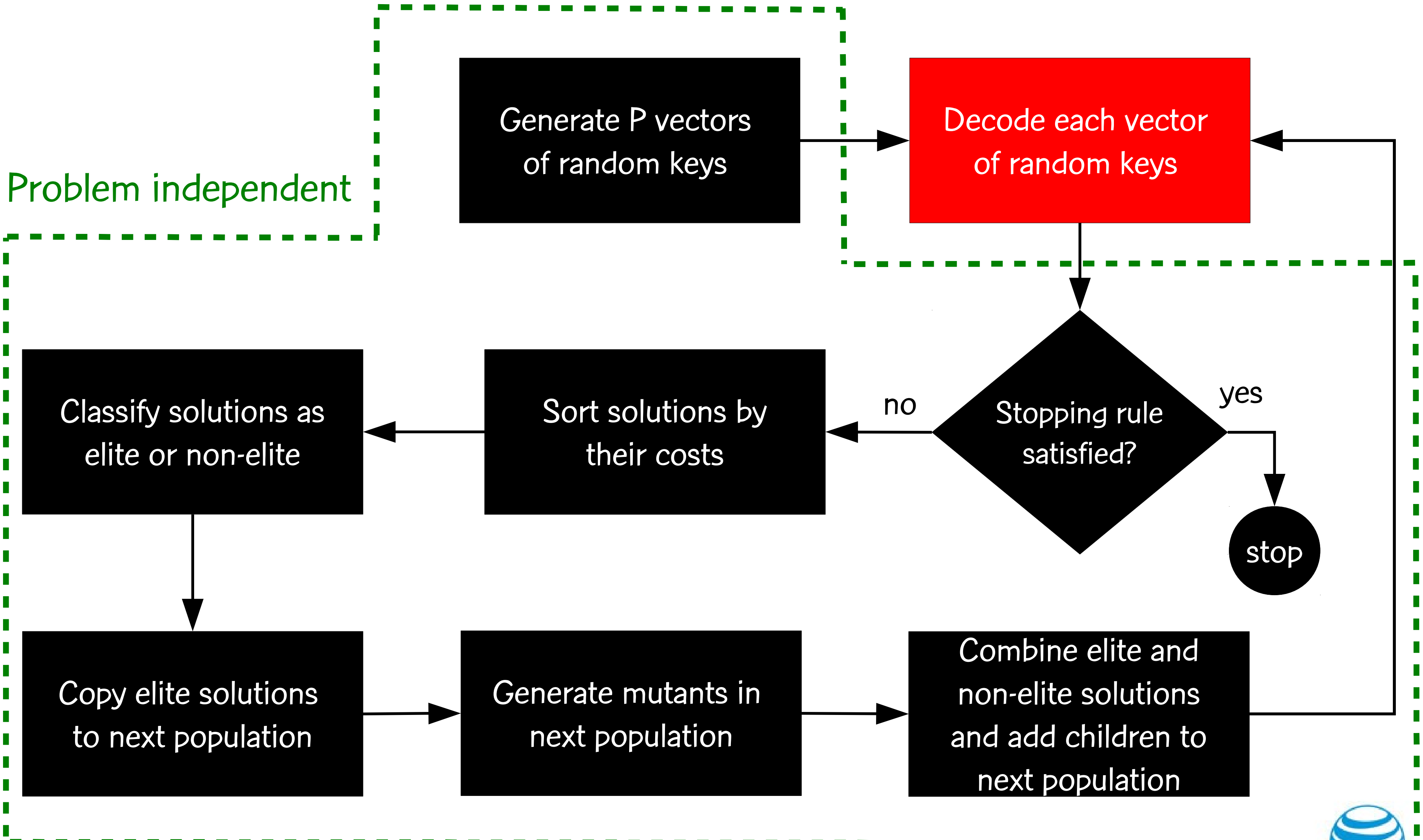
a = (0.25, 0.19, 0.67, 0.05, 0.89)
b = (0.63, 0.90, 0.76, 0.93, 0.08)
c = (0.25, 0.90, 0.76, 0.05, 0.89)

If every random-key array corresponds to a feasible solution: Mating always produces feasible offspring.

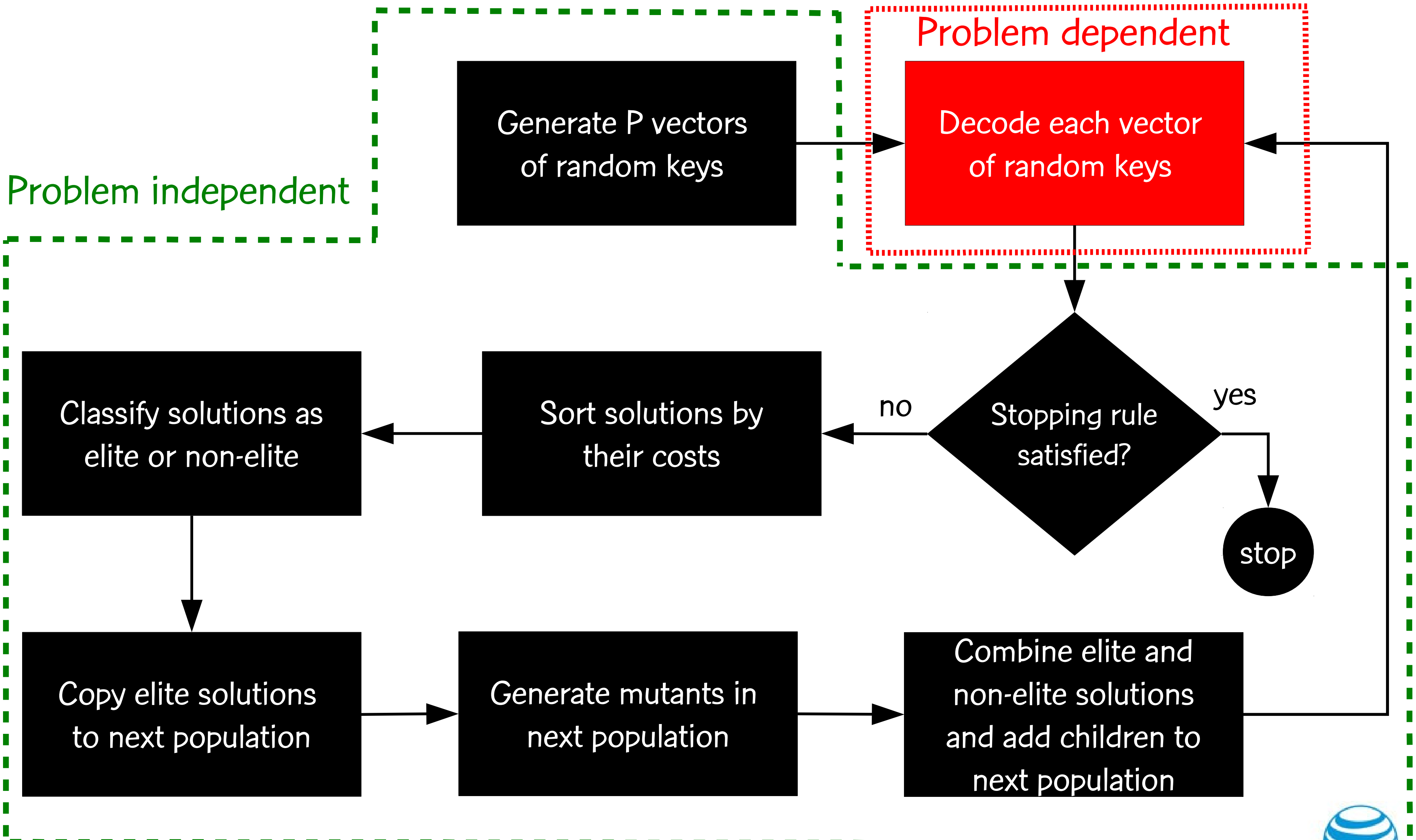
Framework for biased random-key genetic algorithms



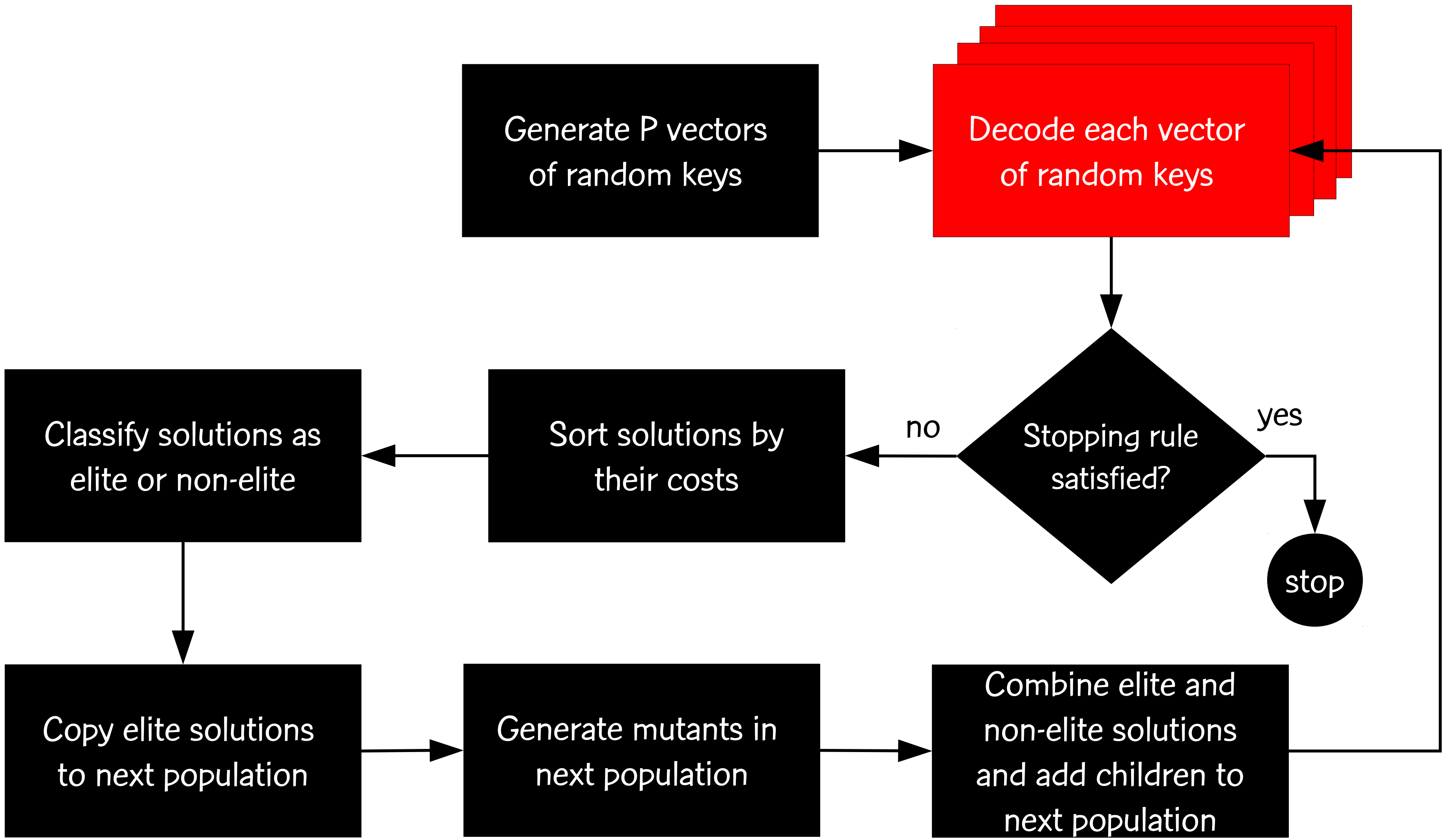
Framework for biased random-key genetic algorithms



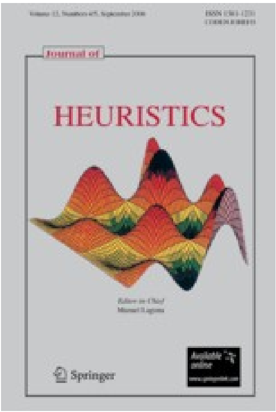
Framework for biased random-key genetic algorithms



Decoding of random key vectors can be done in parallel



Biased random-key genetic algorithms



J.F. Gonçalves and M.G.C.R., "Biased random-key genetic algorithms for combinatorial optimization," J. of Heuristics, vol.17, pp. 487-525, 2011.

Tech report version:

http://mauricio.resende.info/doc/srkga.pdf

Specifying a BRKGA

Specifying a biased random-key GA

- Encoding is always done the same way, i.e. with a vector of N random-keys (parameter N must be specified)

Specifying a biased random-key GA

- Encoding is always done the same way, i.e. with a vector of N random-keys (parameter N must be specified)
- Decoder that takes as input a vector of N random-keys and outputs the corresponding solution of the combinatorial optimization problem and its cost (this is usually a heuristic)

Specifying a biased random-key GA

- Encoding is always done the same way, i.e. with a vector of N random-keys (parameter N must be specified)
- Decoder that takes as input a vector of N random-keys and outputs the corresponding solution of the combinatorial optimization problem and its cost (this is usually a heuristic)
- Parameters

Specifying a biased random-key GA

Parameters:

- Size of population
- Size of elite partition
- Size of mutant set
- Child inheritance probability
- Stopping criterion

Specifying a biased random-key GA

Parameters:

- Size of population: a function of N, say N or 2N
- Size of elite partition
- Size of mutant set
- Child inheritance probability
- Stopping criterion

Specifying a biased random-key GA

Parameters:

- Size of population: a function of N, say N or 2N
- Size of elite partition: 15-25% of population
- Size of mutant set
- Child inheritance probability
- Stopping criterion

Specifying a biased random-key GA

Parameters:

- Size of population: a function of N, say N or 2N
- Size of elite partition: 15-25% of population
- Size of mutant set: 5-15% of population
- Child inheritance probability
- Stopping criterion

Specifying a biased random-key GA

Parameters:

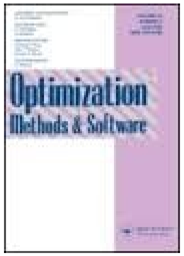
- Size of population: a function of N, say N or 2N
- Size of elite partition: 15-25% of population
- Size of mutant set: 5-15% of population
- Child inheritance probability: > 0.5, say 0.7
- Stopping criterion

Specifying a biased random-key GA

Parameters:

- Size of population: a function of N, say N or 2N
- Size of elite partition: 15-25% of population
- Size of mutant set: 5-15% of population
- Child inheritance probability: > 0.5, say 0.7
- Stopping criterion: e.g. time, # generations, solution quality, # generations without improvement

brkgaAPI: A C++ API for BRKGA



Paper: Rodrigo F. Toso and M.G.C.R.,
"A C++ Application Programming Interface for Biased Random-Key Genetic Algorithms,"
Optimization Methods & Software, published online 13 March 2014.

Software: http://mauricio.resende.info/src/brkgaAPI

The unequal area facility layout problem

Unequal area facility layout

Given

- N rectangular facilities, $i = 1, 2, \ldots, N$, each having given area $A_i = w_i \times h_i$ all of maximum aspect ratio (between longest & shortest dimensions) R (Note that w_i and h_i are not given, only A_i and R are given)

Layout the facilities, without overlap or rotation, on a rectangular floor of area $W \times H$ with centroids at coordinates $(x_1,y_1), (x_2,y_2), \ldots, (x_N,y_N)$ and dimensions $w_1 \times h_1, w_2 \times h_2, \ldots, w_N \times h_N$.

Unequal area facility layout

We consider two types of problems

- In the constrained type, we are given the rectangular floor dimensions $W \times H$.
- In the unconstrained type, we assume the floor space can include all the facilities laid out horizontally or vertically at their maximum horizontal or vertical dimensions, i.e.

$$(W, H) = \left(\sum_{i=1}^{N} (A_i \times R)^{1/2}, \sum_{i=1}^{N} (A_i \times R)^{1/2} \right)$$

Unequal area facility layout

Of all feasible layouts, find one that minimizes

$$\sum_{i=1}^{N} \sum_{j=1}^{N} f_{i,j} \times c_{i,j} \times d_{i,j}$$

where

- $f_{i,j}$ is the flow between facilities i and j ($f_{i,i} = 0$)
- $c_{i,j}$ is the cost per unit distance between i and j
- $d_{i,j} = |x_i - x_j| + |y_i - y_j|$ is the rectilinear distance between (x_i,y_i) and (x_j,y_j)

Unequal area facility layout

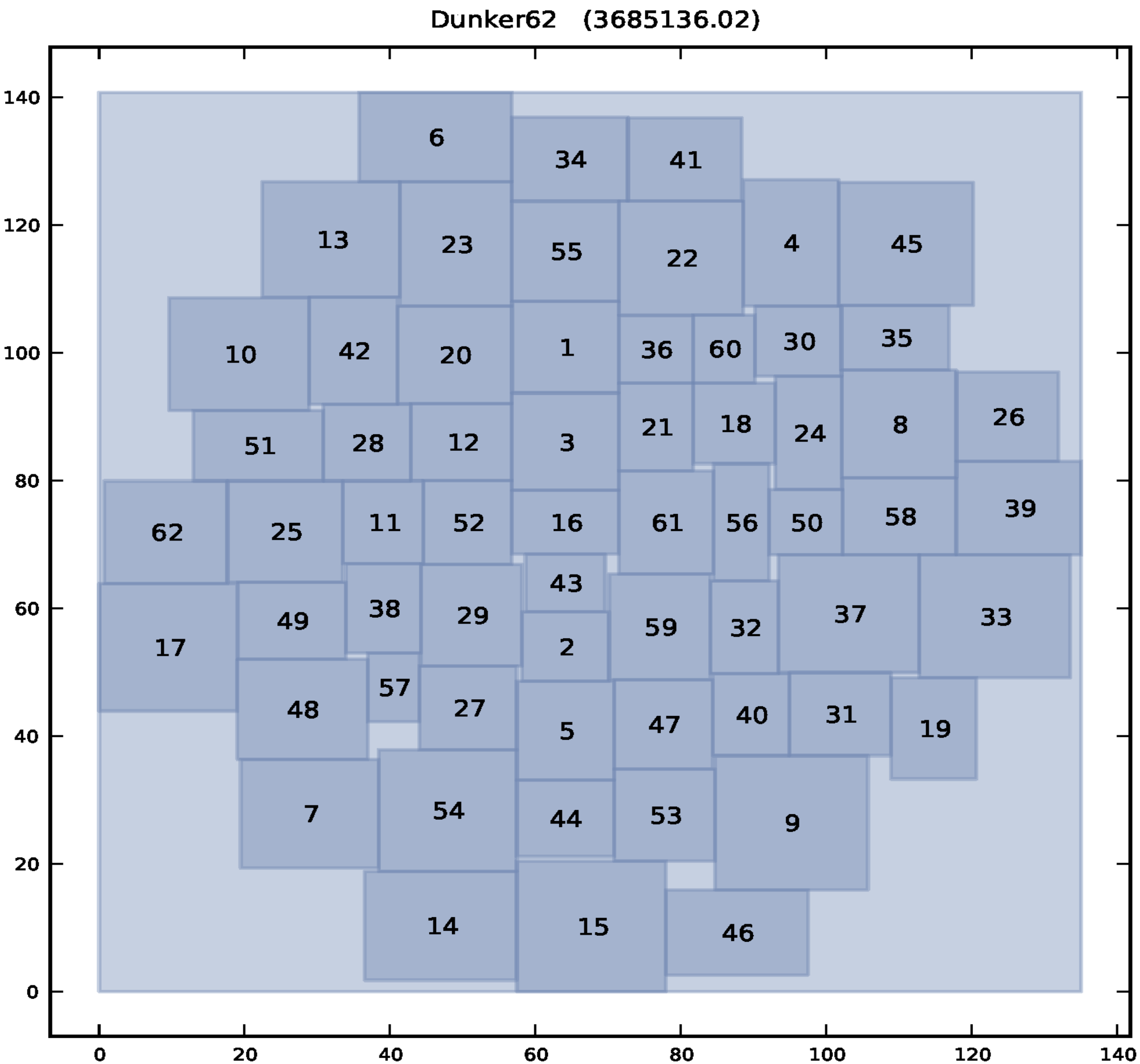
Of all feasible layouts, find one that minimizes

$$\sum_{i=1}^{N} \sum_{j=1}^{N} f_{i,j} \times c_{i,j} \times d_{i,j}$$

Besides rectilinear (R) distance metric, we also deal with Euclidean (E), and Squared Euclidean (SE) in paper.

where

- $f_{i,j}$ is the flow between facilities i and j ($f_{i,i} = 0$)
- $c_{i,j}$ is the cost per unit distance between i and j
- $d_{i,j} = |x_i - x_j| + |y_i - y_j|$ is the rectilinear distance between (x_i,y_i) and (x_j,y_j)

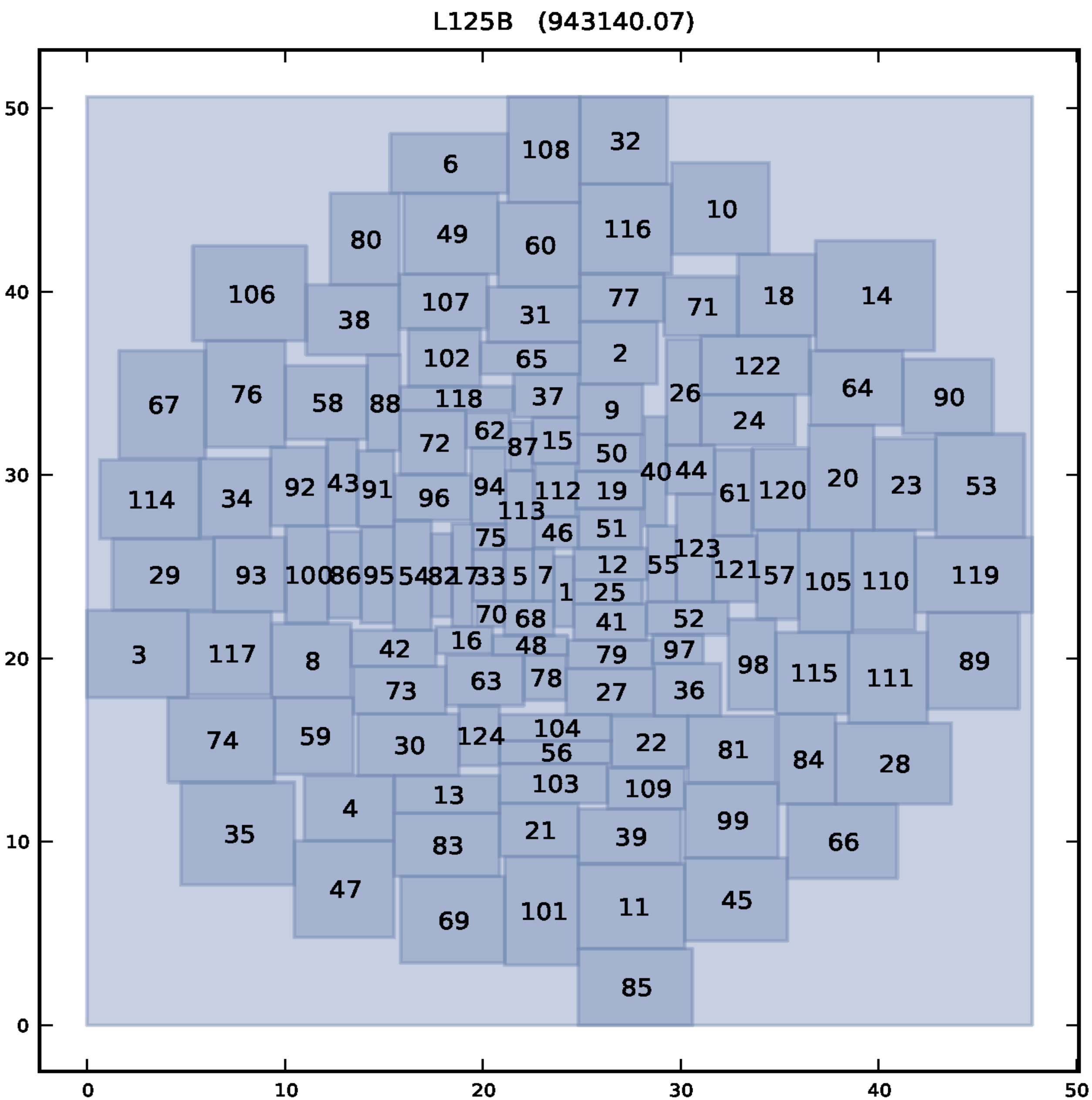


Dunker62

New best known solution: 3.68E6

Previous best known solution: 3.81E6

TS-BST (McKendall Jr. & Hajobyan, 2010)



L125B

New best known solution: 9.43E5

Previous best known solution: 1.01E6

TS-BST (McKendall Jr. & Hajobyan, 2010)

BRKGA for the unequal area facility layout problem

Encoding

Solutions are encoded with a vector of random keys of length 2N+2

$$X = (X_1, \ldots, X_N, X_{N+1}, \ldots, X_{2N}, X_{2N+1}, X_{2N+2})$$

Encoding

Solutions are encoded with a vector of random keys of length 2N+2

$$X = (\underbrace{X_1, \ldots, X_N}, X_{N+1}, \ldots, X_{2N}, X_{2N+1}, X_{2N+2})$$

Facility placement sequence

Encoding

Solutions are encoded with a vector of random keys of length 2N+2

$$X = (\underbrace{X_1, \ldots, X_N}, \underbrace{X_{N+1}, \ldots, X_{2N}}, X_{2N+1}, X_{2N+2})$$

Facility placement sequence

Facility aspect ratios

Encoding

Solutions are encoded with a vector of random keys of length 2N+2

$$X = (\underbrace{X_1, \ldots, X_N,}_{\text{Facility placement sequence}} \; \underbrace{X_{N+1}, \ldots, X_{2N},}_{\text{Facility aspect ratios}} \; \underbrace{X_{2N+1}, X_{2N+2}}_{\text{(x, y) coordinates of the first facility to be placed}})$$

Decoding

1. Use $X_1, \ldots, X_N$ to determine the sequence in which the facilities are placed on the floor space
2. Use $X_{N+1}, \ldots, X_{2N}$ to determine the aspect ratio of each facility
3. Use X_{2N+1}, X_{2N+2} to determine the (x, y) coordinates of the first facility to be placed on the floor space
4. Use results of (1)-(3) with placement heuristic to place all the facilities on the floor space
5. Evaluate fitness of solution

Decoder: Step 1

Use $X_1, \ldots, X_N$ to determine the sequence in which the facilities are placed on the floor space:

Simply sort the key values $X_1, \ldots, X_N$ to determine the indices of the permutation of the facilities.

Decoder: Step 2

Use X_{N+1}, ..., X_{2N} to determine the aspect ratio of each facility:

Aspect ratio of facility i is

$$FAR_i = (1/R) + X_{N+i} \times (R - (1/R)),$$

where R is the given maximum facility aspect ratio.

Decoder: Step 2

Use $X_{N+1}, \ldots, X_{2N}$ to determine the aspect ratio of each facility:

Aspect ratio of facility i is

$$FAR_i = (1/R) + X_{N+i} \times (R - (1/R)),$$

where R is the given maximum facility aspect ratio.

Since $FAR_i = w_i/h_i$, then

$w_i = (A_i \times FAR_i)^{1/2}$ and

$h_i = A_i/w_i$

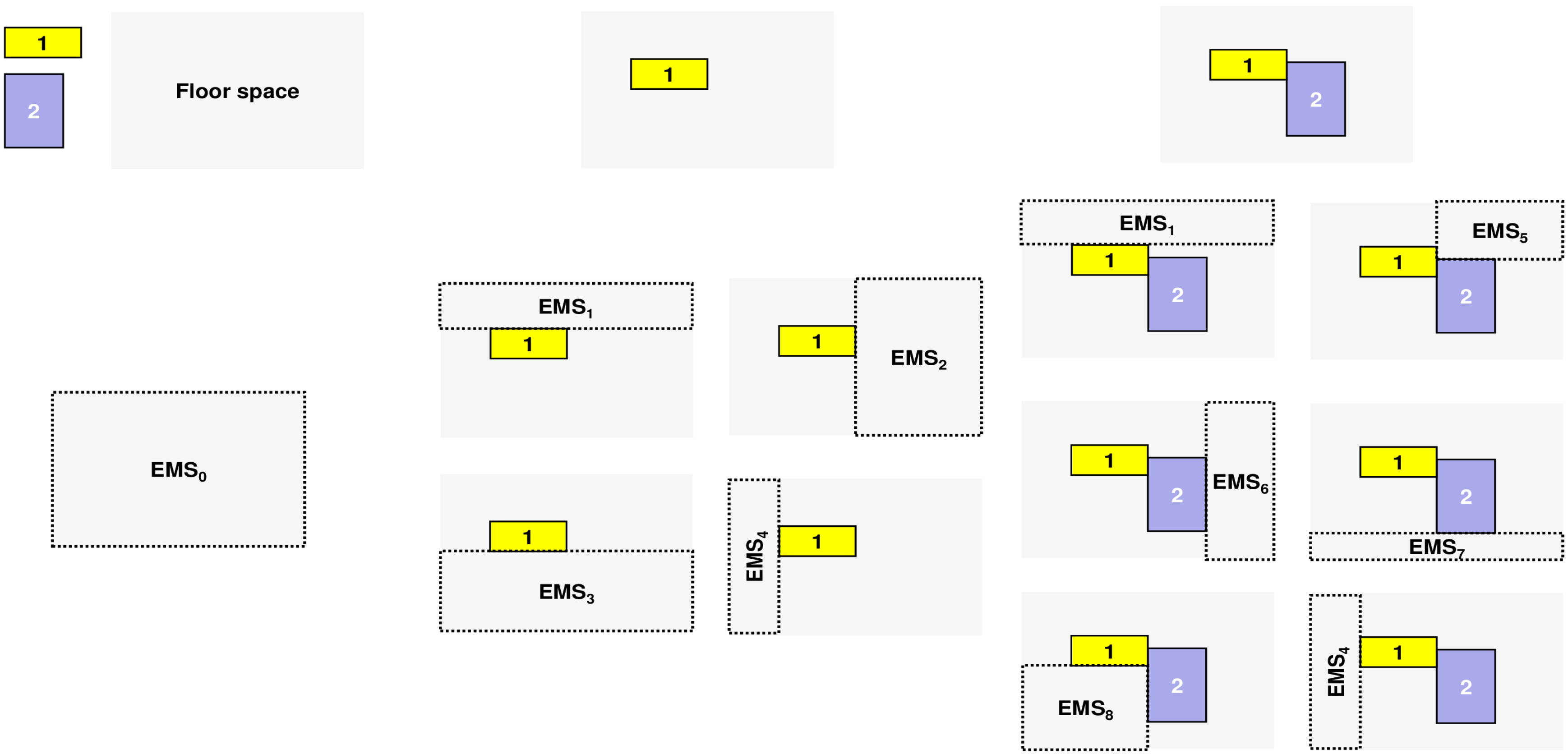
Decoder: Step 3

Use X_{2N+1}, X_{2N+2} to determine the (x, y) coordinates of the first facility to be placed on the floor space.

$$x = (w_i/2) + X_{2N+1} \times (W - w_i)$$

$$y = (h_i/2) + X_{2N+2} \times (H - h_i)$$

Decoder: Step 4 Makes use of empty maximal-spaces (EMS)



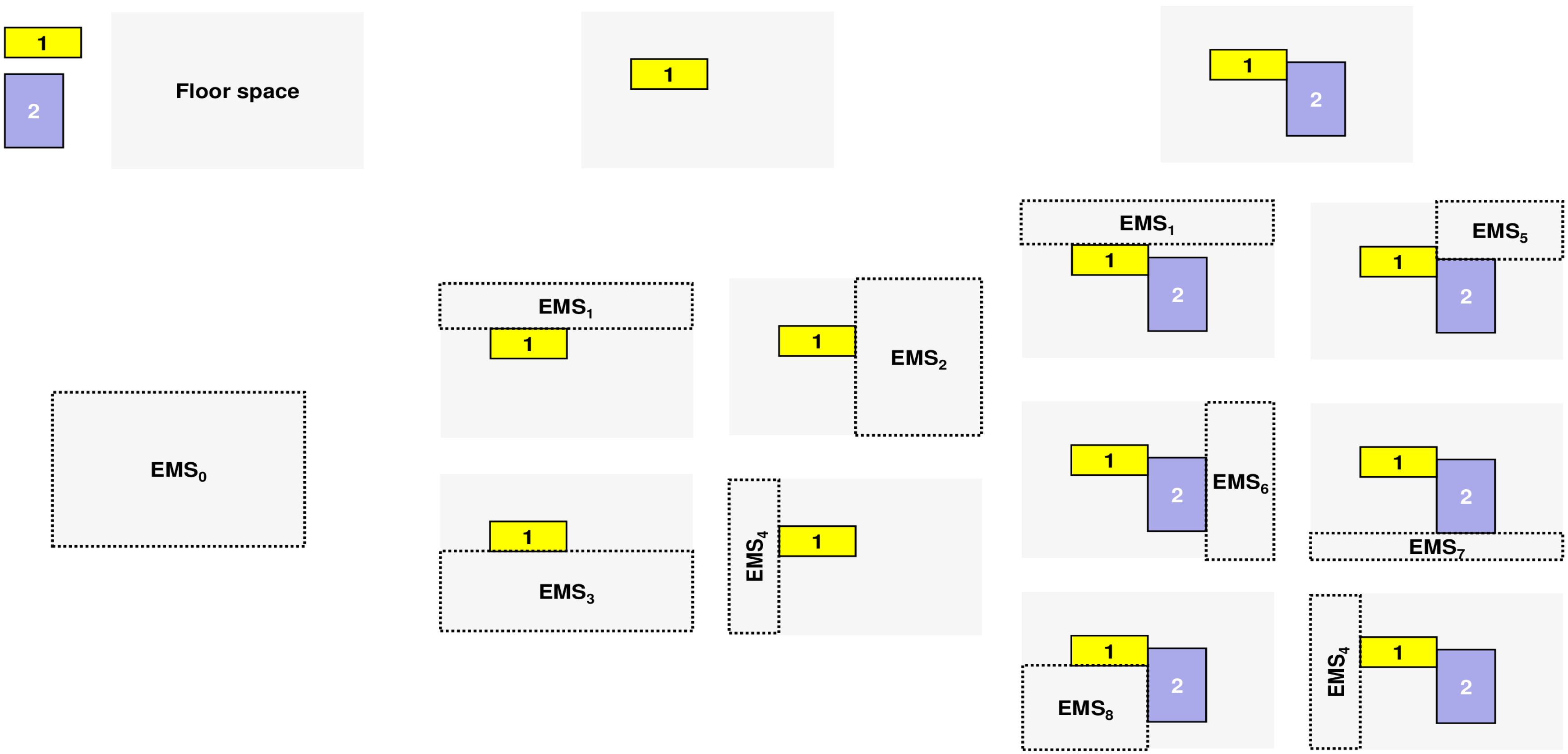
a) Facilities to be placed and the initial empty maximal-space (the floor space)

b) Empty maximal-spaces after placing facility 1.

c) Empty maximal-spaces after placing facility 2.

Decoder: Step 4

When placing a facility we only consider EMSs where the facility fits. This way we avoid overlapping.

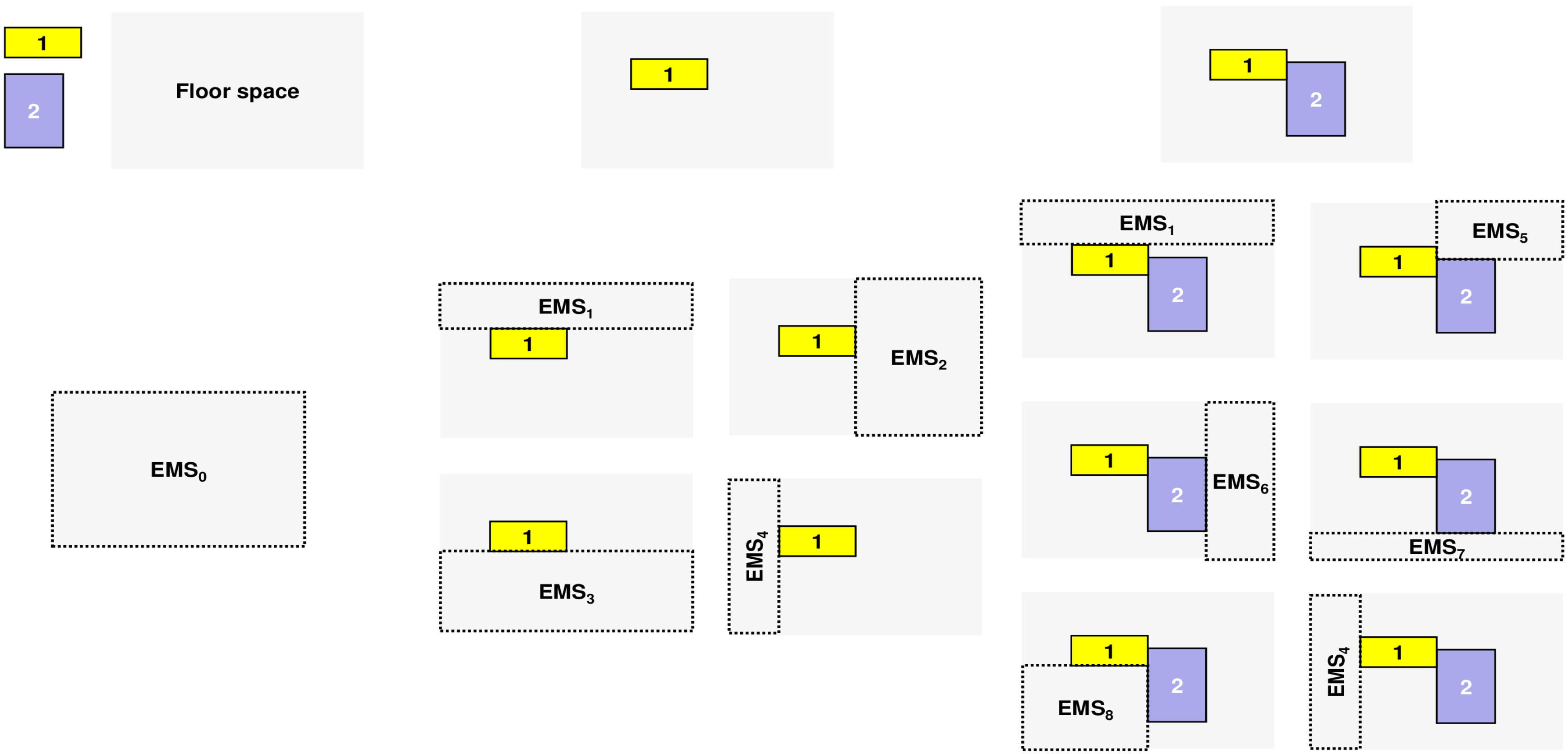


a) Facilities to be placed and the initial empty maximal-space (the floor space)

b) Empty maximal-spaces after placing facility 1.

c) Empty maximal-spaces after placing facility 2.

Decoder: Step 4 EMSs are generated and kept track of with the Difference Process (DP) of Lai and Chan (1997).



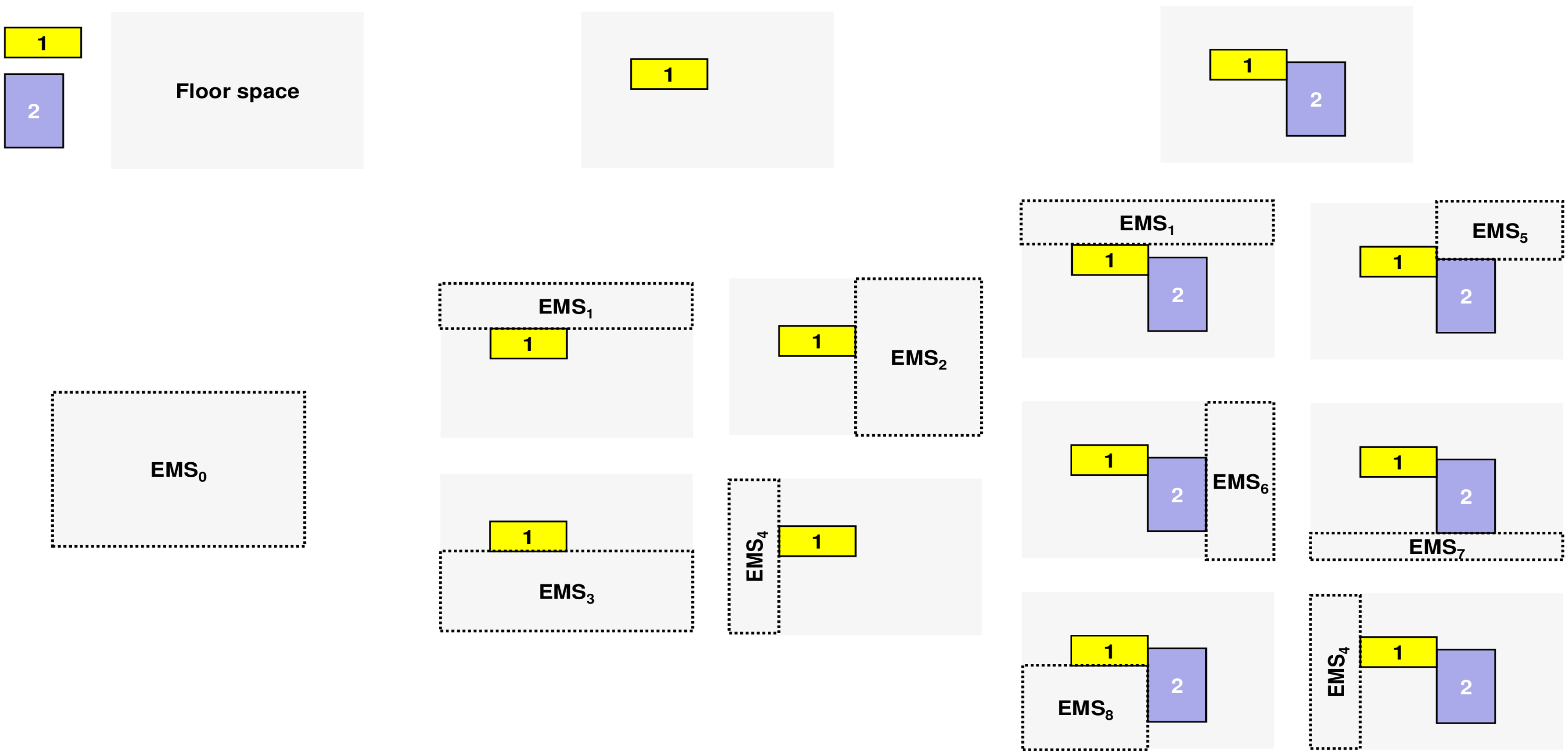
a) Facilities to be placed and the initial empty maximal-space (the floor space)

b) Empty maximal-spaces after placing facility 1.

c) Empty maximal-spaces after placing facility 2.

Decoder: Step 4

Recall that in the unconstrained case the floor space can include all facilities laid out horizontally or vertically.



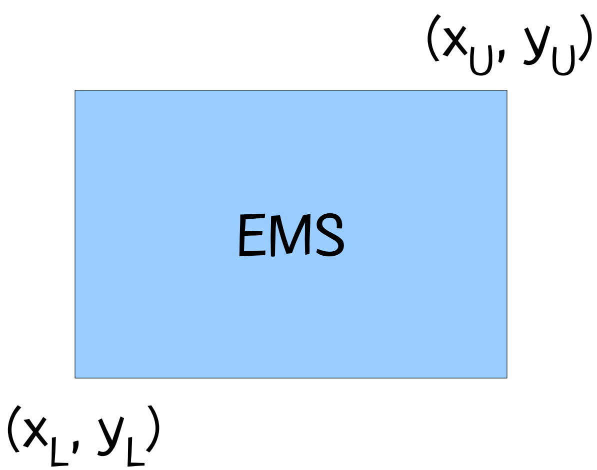
a) Facilities to be placed and the initial empty maximal-space (the floor space)

b) Empty maximal-spaces after placing facility 1.

c) Empty maximal-spaces after placing facility 2.

Decoder: Step 4

For each EMS in which the facility fits, we compute the incremental cost associated with placing the facility in that EMS and then place it in the least-cost EMS.



Compute positions that minimize cost of placing facility i in each available EMS $\{(x_L, y_L), (x_U, y_U)\}$ w.r.t. all already-placed facilities K:

$$\min \sum_{k \in K} c_{i,k} \times f_{i,k} \times d_{i,k}$$

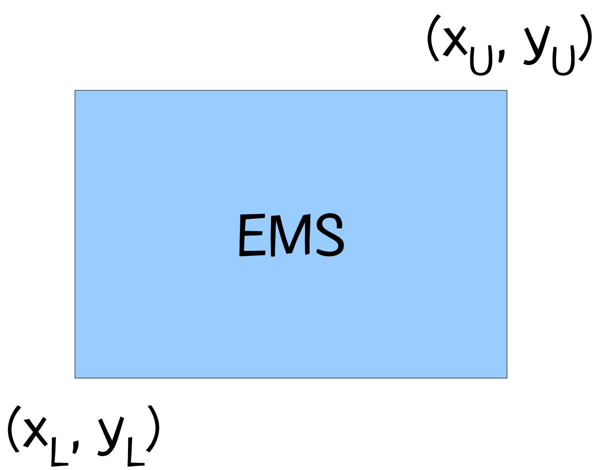
subject to:

$$x_L + w_i/2 \leq x_i \leq x_U - w_i/2$$

$$y_L + h_i/2 \leq y_i \leq y_U - h_i/2$$

Decoder: Step 4

For each EMS in which the facility fits, we compute the incremental cost associated with placing the facility in that EMS and then place it in the least-cost EMS.



Compute positions that minimize cost of placing facility i in each available EMS $\{(x_L, y_L), (x_U, y_U)\}$ w.r.t. all already-placed facilities K:

$$\min \sum_{k \in K} c_{i,k} \times f_{i,k} \times d_{i,k}$$

subject to:

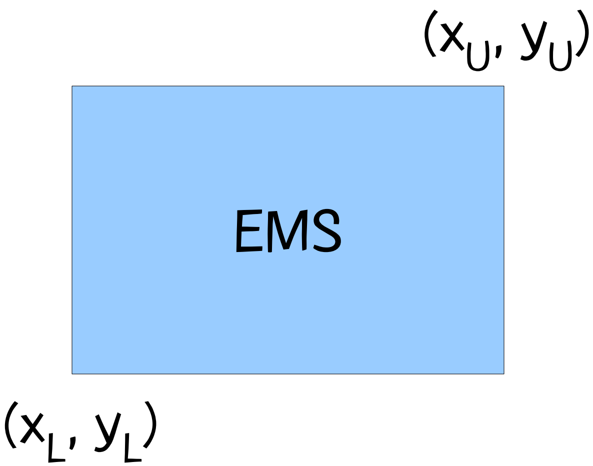
$$x_L + w_i/2 \leq x_i \leq x_U - w_i/2$$

$$y_L + h_i/2 \leq y_i \leq y_U - h_i/2$$

Instead of solving this directly with a NLP solver we propose a different approach.

Decoder: Step 4

For each EMS in which the facility fits, we compute the incremental cost associated with placing the facility in that EMS and then place it in the least-cost EMS.



Find the unconstrained optimum (UO) using a method described in Heragu (1997):

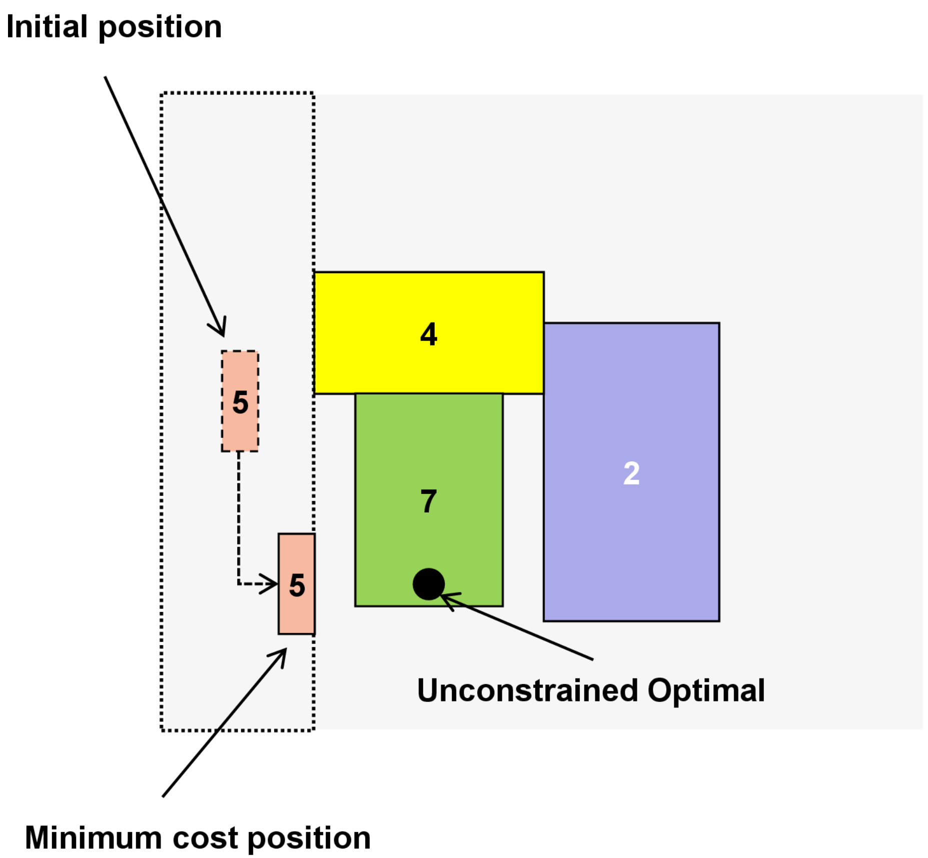
$$\min \sum_{k \in K} c_{i,k} \times f_{i,k} \times d_{i,k}$$

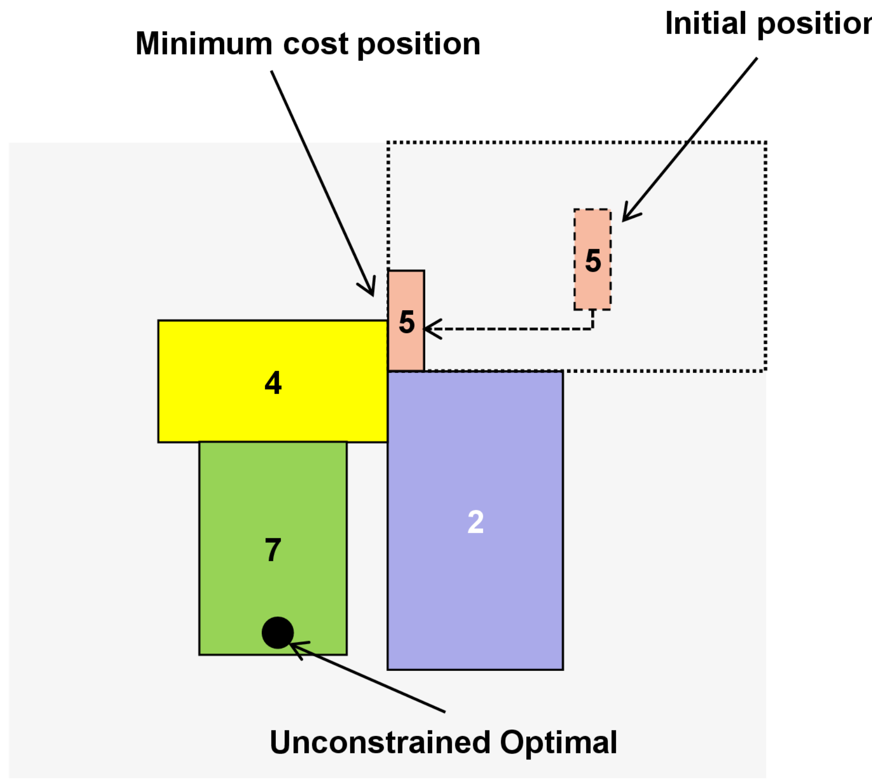
If there is no flow between facility i and the already laid-out facilities, then UO is assumed to be geometric center of all laid-out facilities.

Tentatively place facility i in the geometric center of each EMS in which it fits.

Decoder: Step 4

For each EMS in which the facility fits, we place the facility in the center of the EMS and move it as close as possible to the UO and compute the objective.





Experimental results – Unconstrained

We compare our BRKGA with eight algorithms:

- Hierarchical approach with clusters (HA-C) of Tam and Li (1991)
- GA with slicing tree structure (GA-STS) of Kado (1996)
- Genetic programming algorithm (GP-STS) of Garces-Perez et al. (1996)

Experimental results – Unconstrained

We compare our BRKGA with eight algorithms:

- GA with tree-structured genotype representation (GA-TSG) of Schnecke and Vornberger (1997)
- Tabu search with slicing tree (TSaST) of Scholtz et al. (2009)
- Commercial solver from Engineering Optimization Software (VIP-PLANOPT) based on algorithms of Mir and Imam (1996, 2001) and Imam and Mir (1998)

Experimental results – Unconstrained

We compare our BRKGA with eight algorithms:

- Tabu search with boundary search technique (TS-BST) of McKendall Jr. and Hakobyan (2010)
- The MIP solver from Gurobi Optimization (Gurobi) version 5.5.

Experimental results – Unconstrained

Benchmark instances:

- Seven L instances of Imam and Mir (1993, 1998), Mir and Imam (1996, 2001), and VIP-PLANOPT (2006, 2010) with 20 to 125 facilities
- Dunker62 instance of Dunker et al. (2003) with 62 facilities
- Eight TL instances of Tam and Li (1991) with 5 to 30 instances
- 100 random (RND) instances with known optimal with 10 to 100 facilities of Gonçalves & MGCR (2014)

Experimental results – Unconstrained

Computational setup:

- BRKGA coded in C++
- Experiments run on a computer with an Intel Xeon E5-2630 processor at 2.30 GHz and 16 GB of RAM running Linux O.S. (Fedora, release 18)
- BRKGA parameters
  - Population size: $p = 100 \times N$
  - Elite population: min ($0.25 \times p$, 50)
  - Mutation population: $0.25 \times p$
  - Inheritance probability: 0.70
  - Stopping rule: 50 generations

at&t Your world. Delivered.

Experimental results – Unconstrained

	VIP-PLANOPT		TSaST		TS-BST		BRKGA		
Dataset	Cost	Time	Cost	Time	Cost	Time	Cost	Time	%Impr
L20	1.13E3	0.3	-	-	1.15E3	10351.9	1.13E3	0.5	1.86
L28	6.45E3	1.5	-	-	-	-	6.01E3	1.0	6.72
L50	7.82E4	7.0	-	-	7.13E4	7626.5	6.94E4	6.3	2.65
L75	3.44E4	13.0	-	-	-	-	3.15E4	11.6	8.47
L100	5.38E5	14.0	-	-	4.97E5	11397.2	4.79E5	57.0	3.60
L125A	2.89E5	110.0	-	-	-	-	2.57E5	83.6	11.05
L125B	1.08E6	70.0	-	-	1.01E6	9250.3	9.43E5	118.7	6.51
Dunker62	3.94E6	4996.0	3.87E6	252.0	3.81E6	7304.1	3.69E6	9.1	3.35

Times are in seconds

Experimental results – Unconstrained

	HA-C	GA-STS	GP-STS	GA-TSG	TSaST		BRKGA		
Dataset	Cost	Cost	Cost	Cost	Cost	Time	Cost	Time	%Impr
TL05	247	228	226	214	213.5	2.3	210.1	0.035	1.60
TL06	514	361	384	327	348.8	3.0	345.0	0.049	(5.51)
TL07	559	596	568	629	562.9	2.5	549.7	0.060	1.67
TL08	839	878	878	833	810.4	4.7	799.1	0.080	1.40
TL12	3162	3283	3220	3164	3054.2	12.5	2920.5	0.162	4.38
TL15	5862	7384	7510	6813	6615.8	17.0	6395.4	0.251	(9.10)
TL20	-	16393	14033	13190	13198.4	50.0	9892.4	0.443	25.00
TL30	-	41095	39018	25358	33721.5	95.4	31454.2	1.132	6.72

Times are in seconds

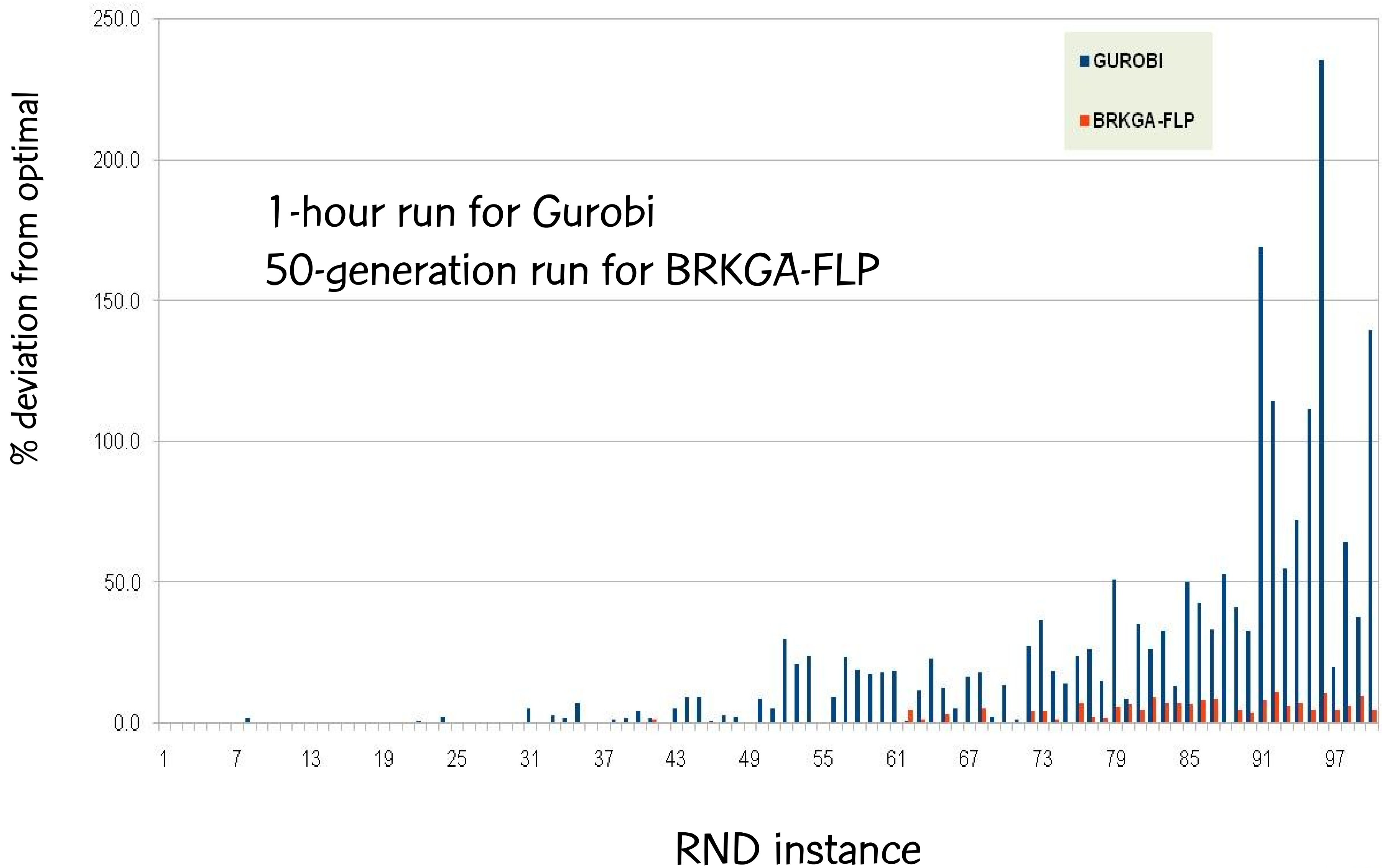
Experimental results – Unconstrained

Each dataset consists of 10 instances, each with known optimum.

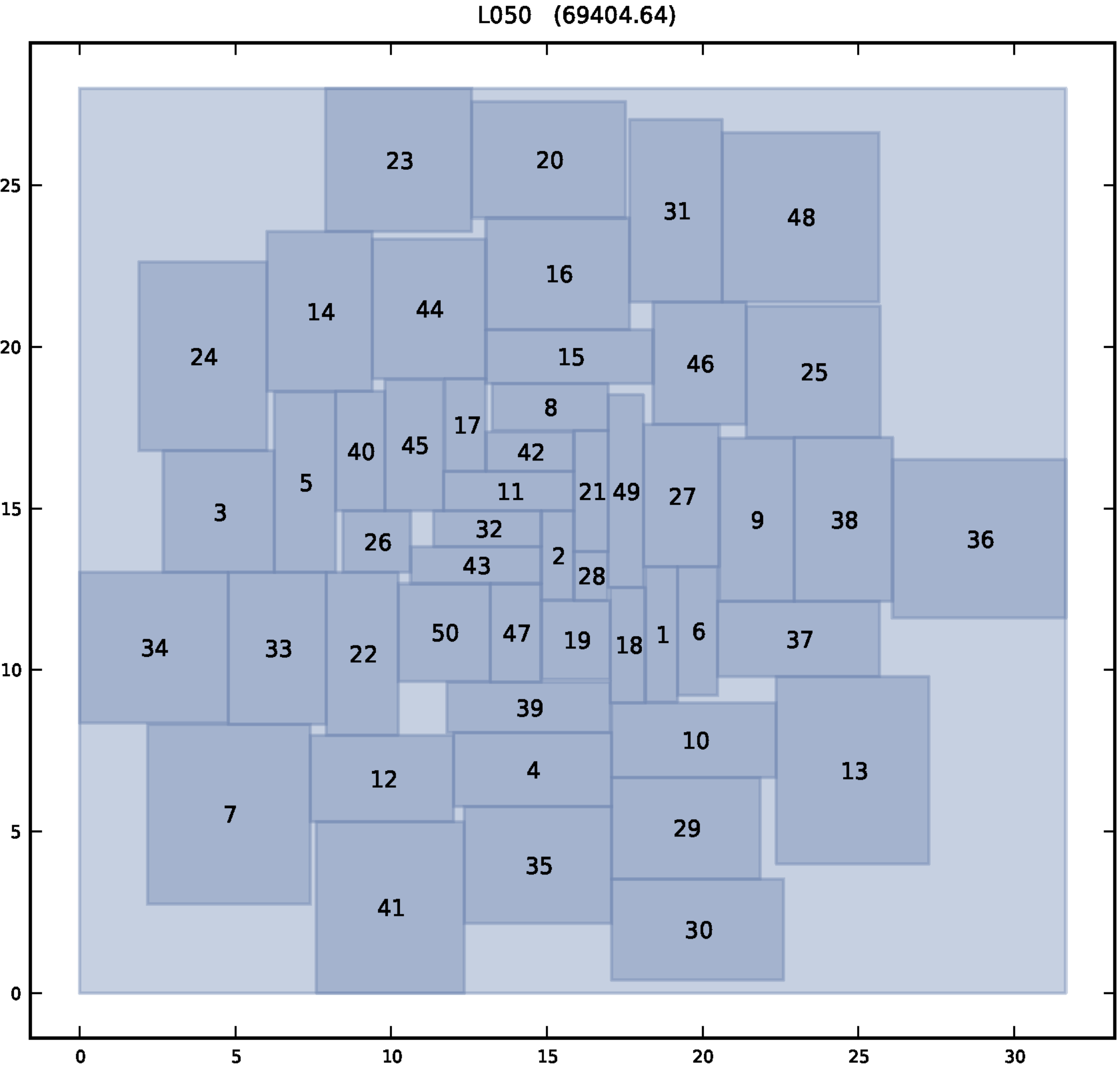
	Gurobi			BRKGA		
Dataset	**Time**	**Avg % Dev**	**Max % Dev**	**Time**	**Avg % Dev**	**Max % Dev**
RND10	3600	0.21	1.66	1.76	0.00	0.00
RND20	3600	0.01	0.12	6.13	0.00	0.00
RND30	3600	0.32	2.14	15.00	0.00	0.00
RND40	3600	2.37	7.10	28.67	0.00	0.00
RND50	3600	3.99	9.30	48.30	0.11	1.12
RND60	3600	16.65	29.73	72.86	0.02	0.15
RND70	3600	12.21	22.70	102.90	1.44	5.29
RND80	3600	22.31	50.97	143.37	3.31	7.10
RND90	3600	36.11	52.99	186.87	6.00	9.09
RND100	3600	101.78	235.31	235.84	7.36	10.97

Times are in seconds

% deviation from optimum



% deviation from optimal
250.0
200.0
150.0
100.0
50.0
0.0
GUROBI
BRKGA-FLP
1-hour run for Gurobi
50-generation run for BRKGA-FLP
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
RND instance

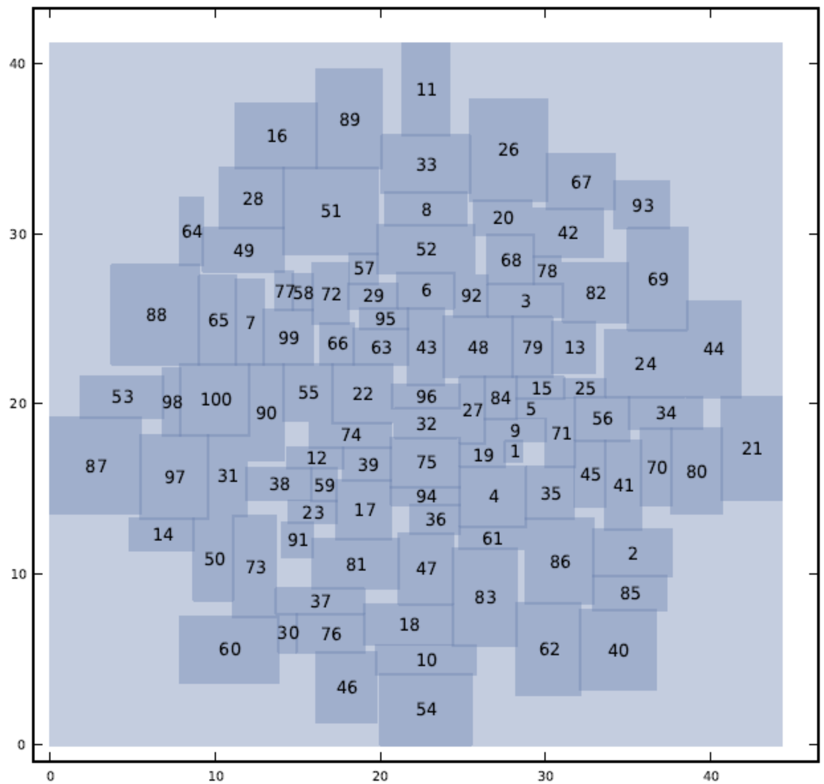


L050

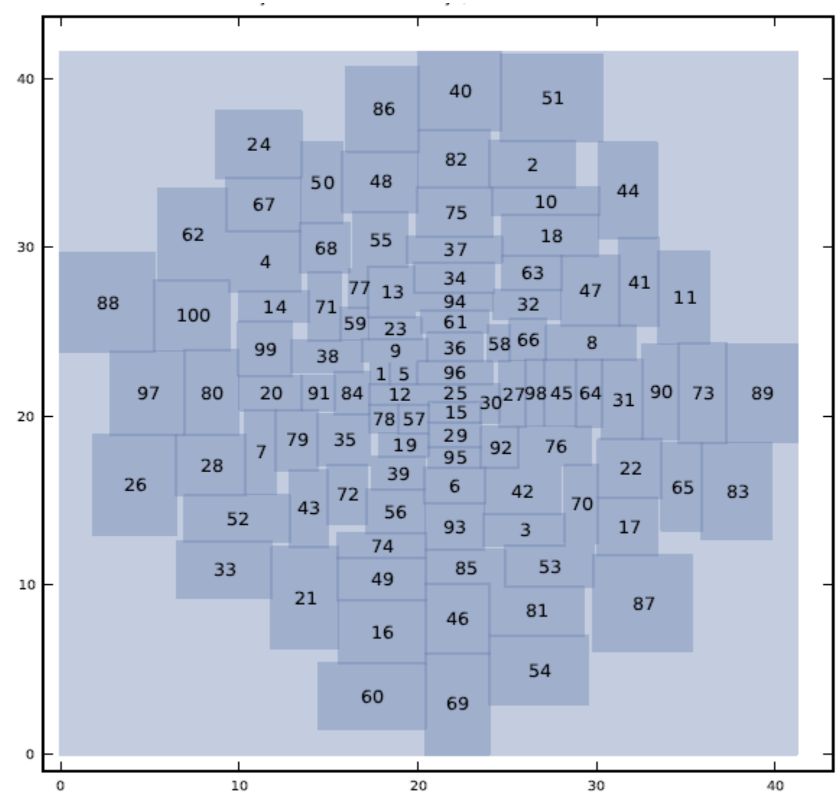
New best known Solution: 6.94E4

Previous best known Solution: 7.13E4 TS-BST (McKendall Jr. & Hajobyan, 2010)

1st generation: 530404.76

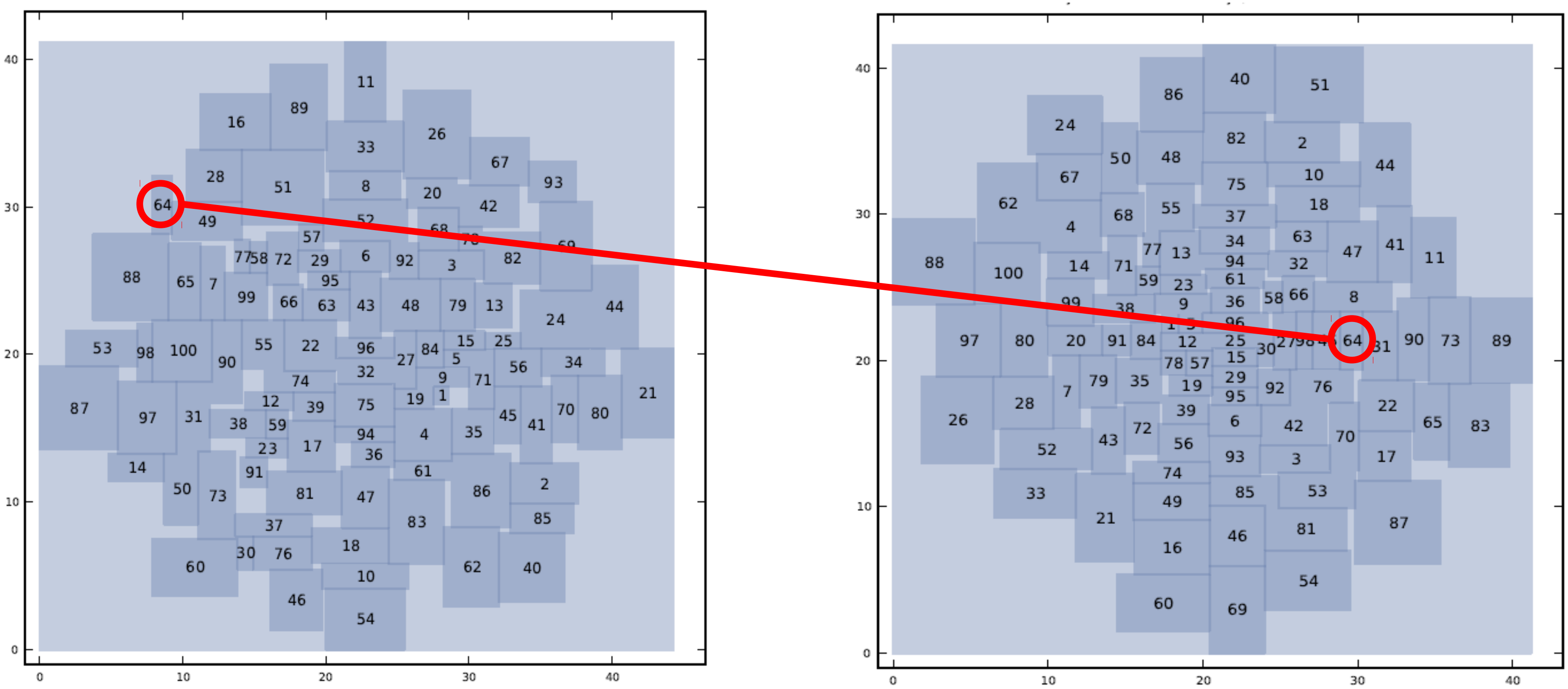


50th generation: 478910.09



L100

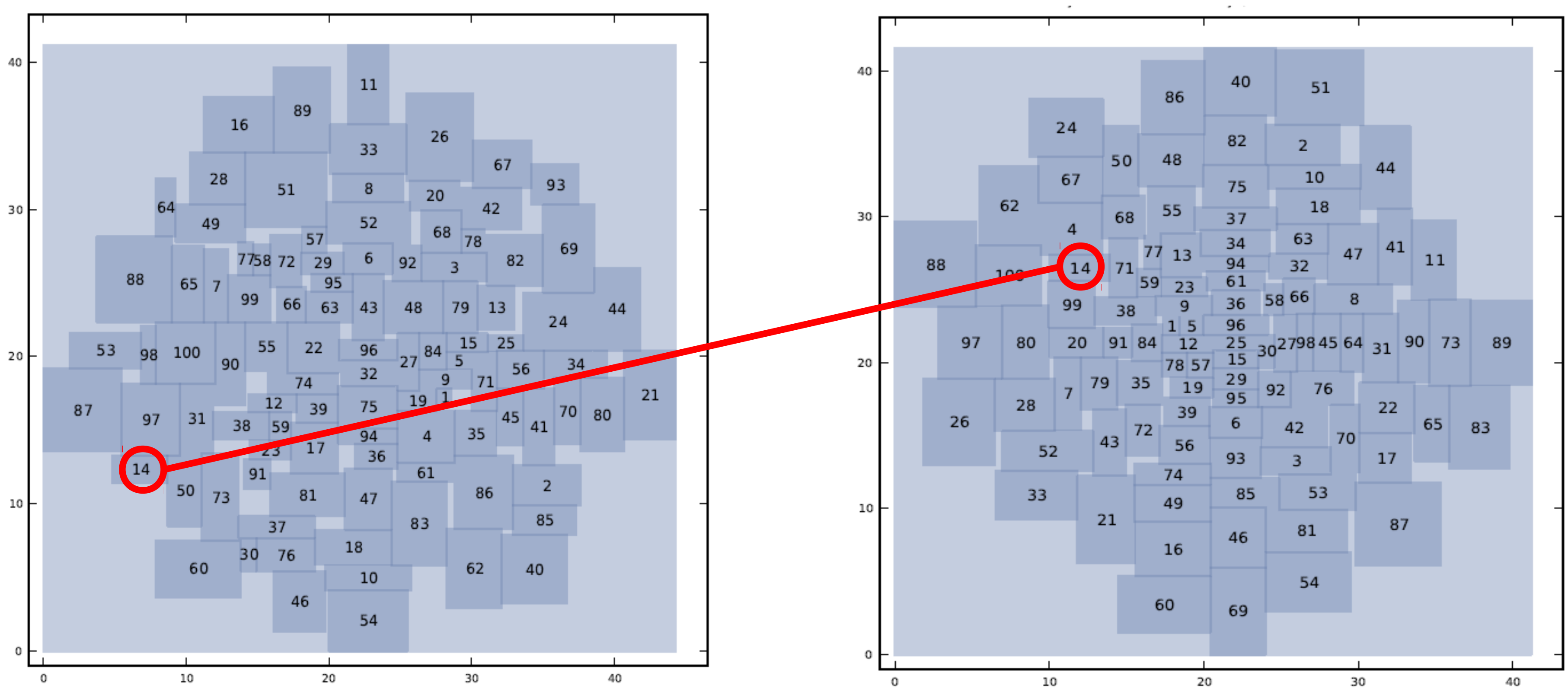
1st generation: 530404.76 | 50th generation: 478910.09



L100

1st generation: 530404.76

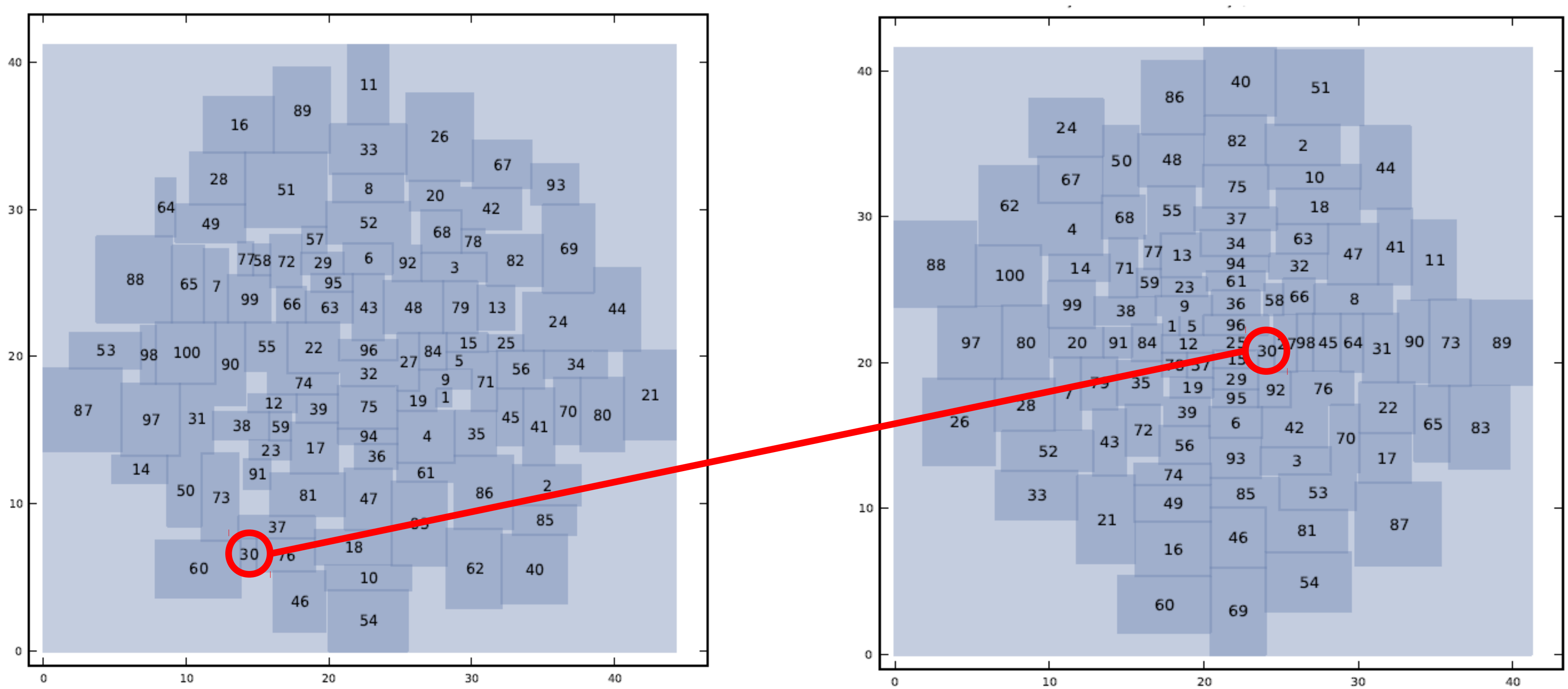
50th generation: 478910.09



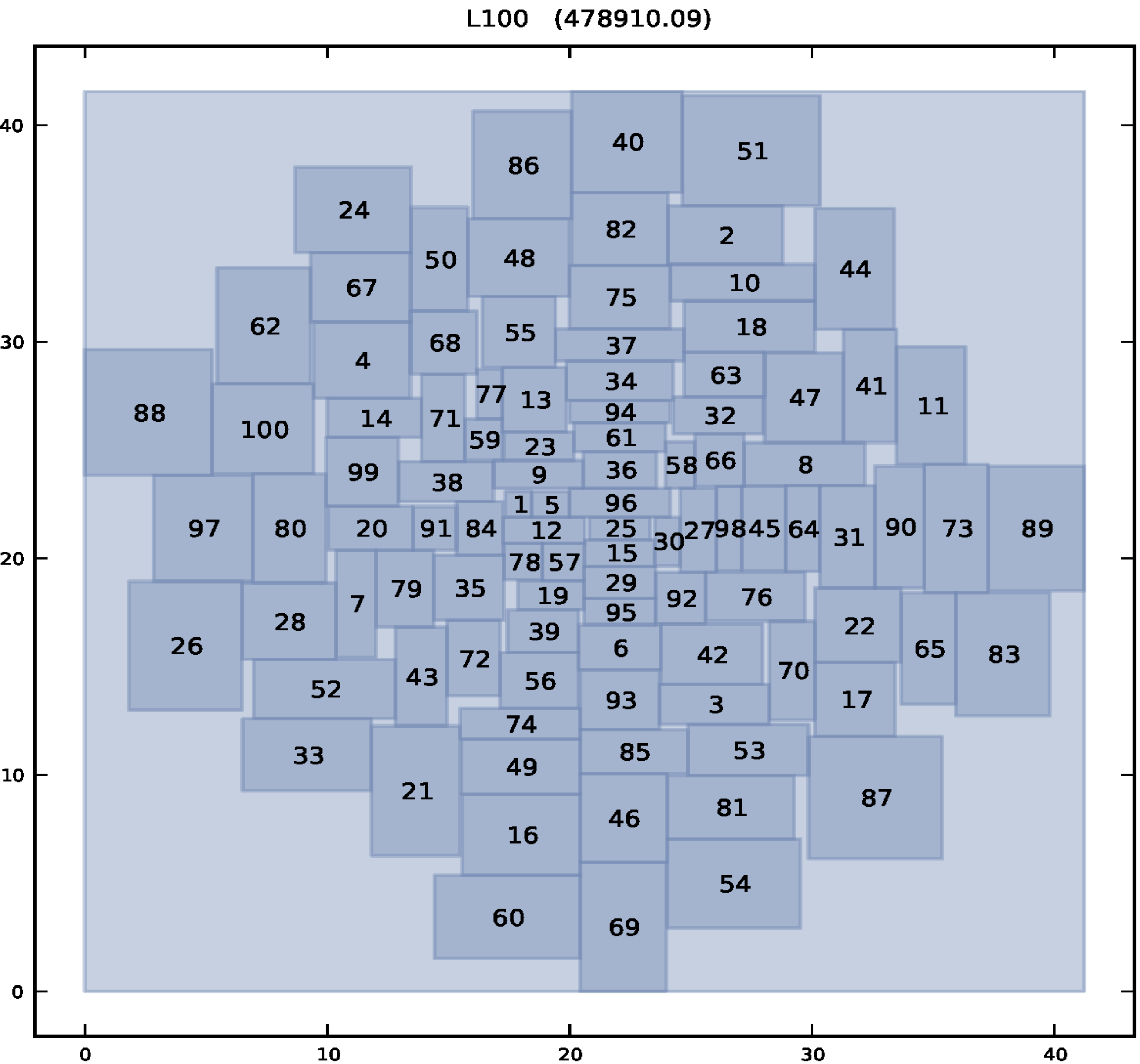
L100

1st generation: 530404.76

50th generation: 478910.09



L100

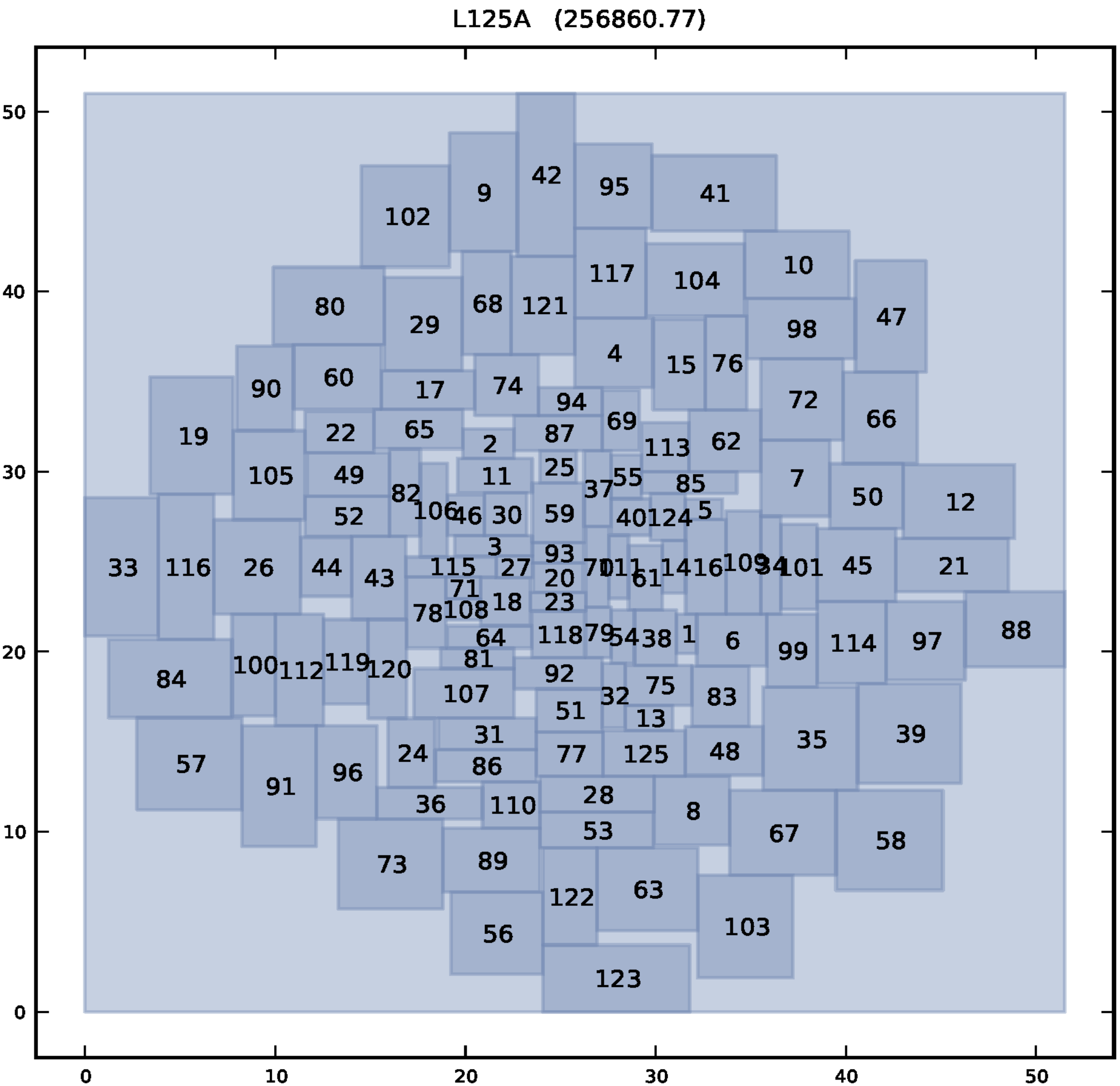


L100

New best known Solution: 4.79E5

Previous best known Solution: 4.97E5

TS-BST (McKendall Jr. & Hajobyan, 2010)



L125A

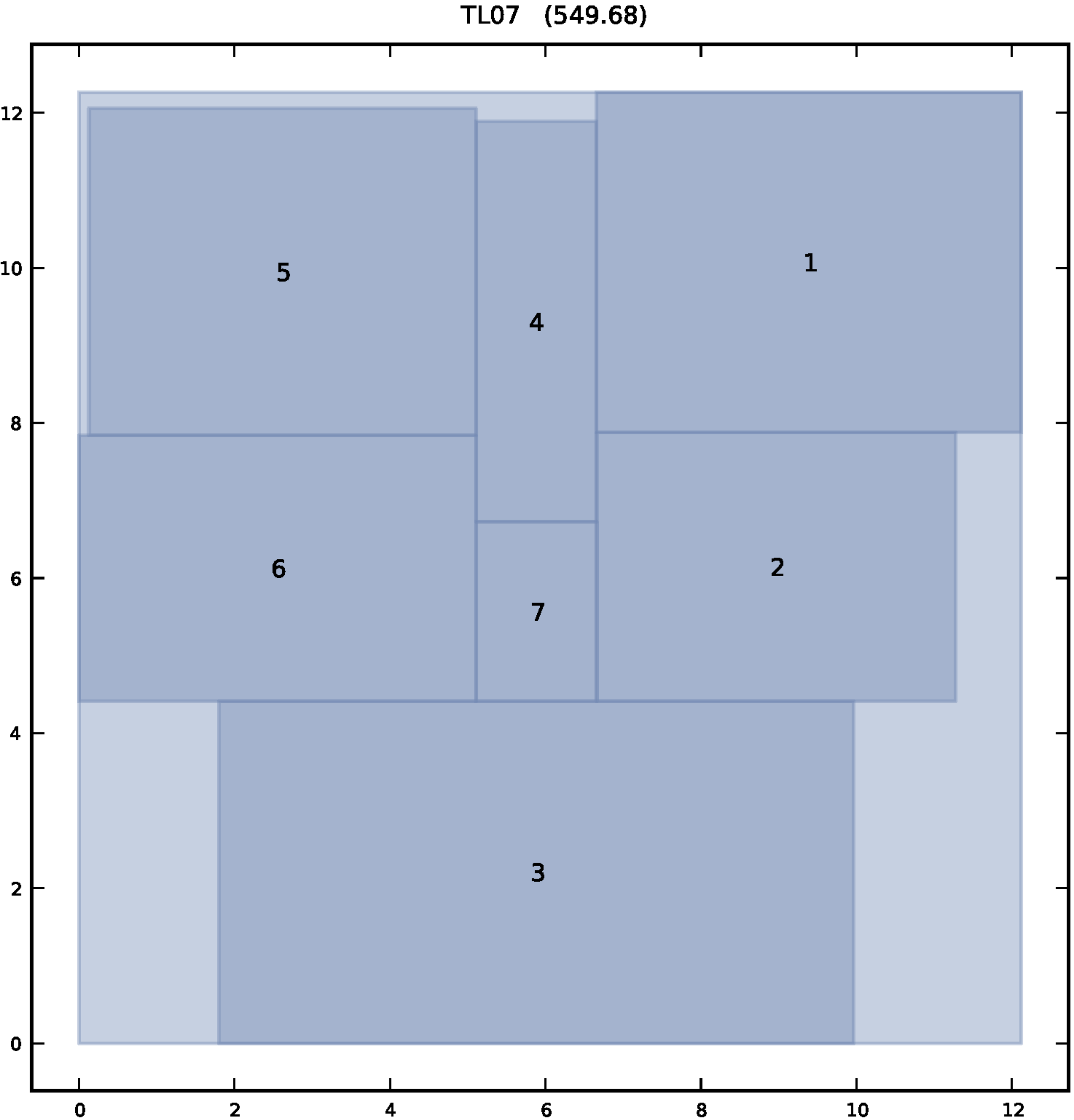
New best known

Solution: 2.57E5

Previous best known

Solution: 2.89E5

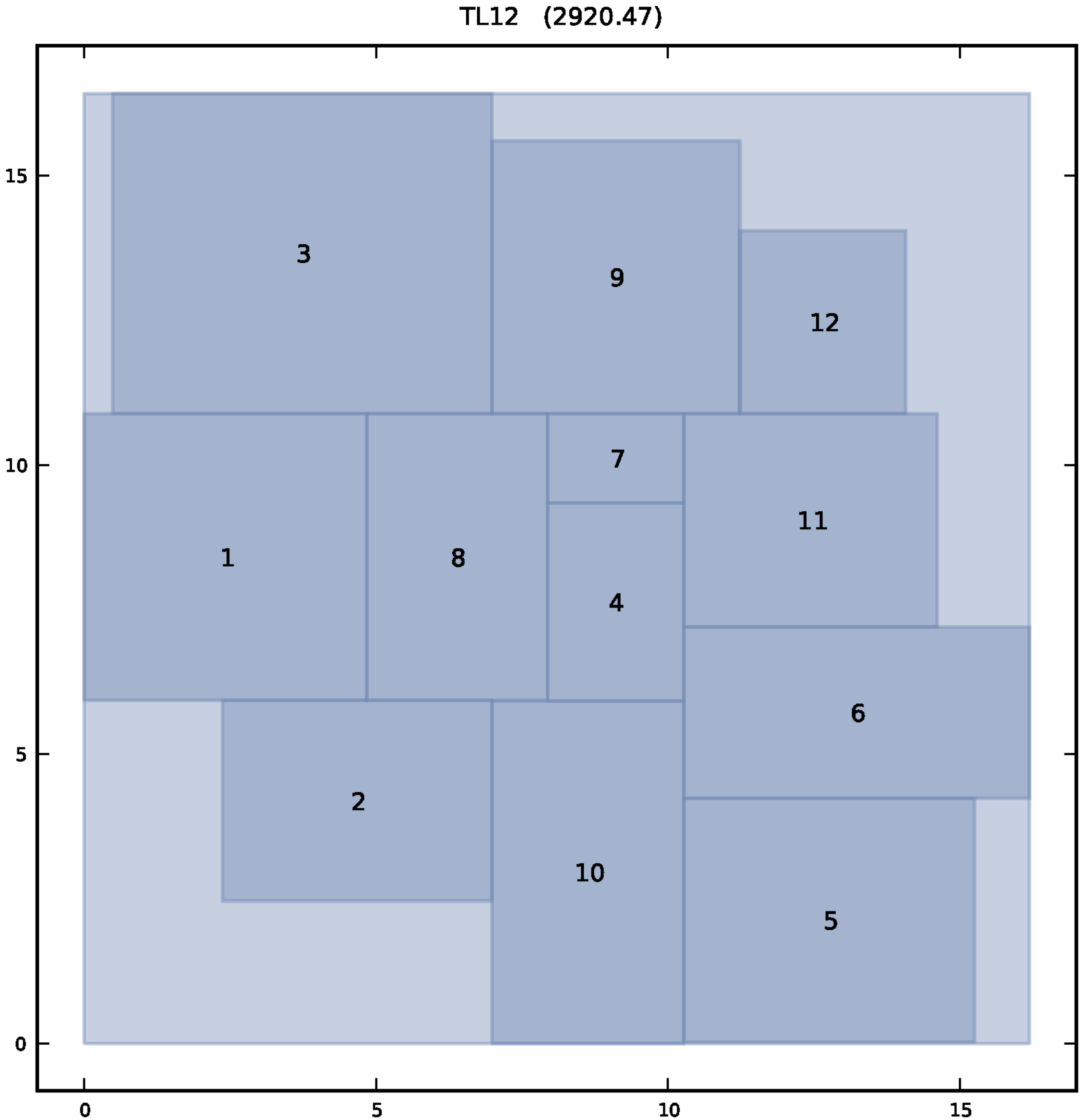
VIP-PLANOPT (2010)



TL07

New best known
Solution: 549.7

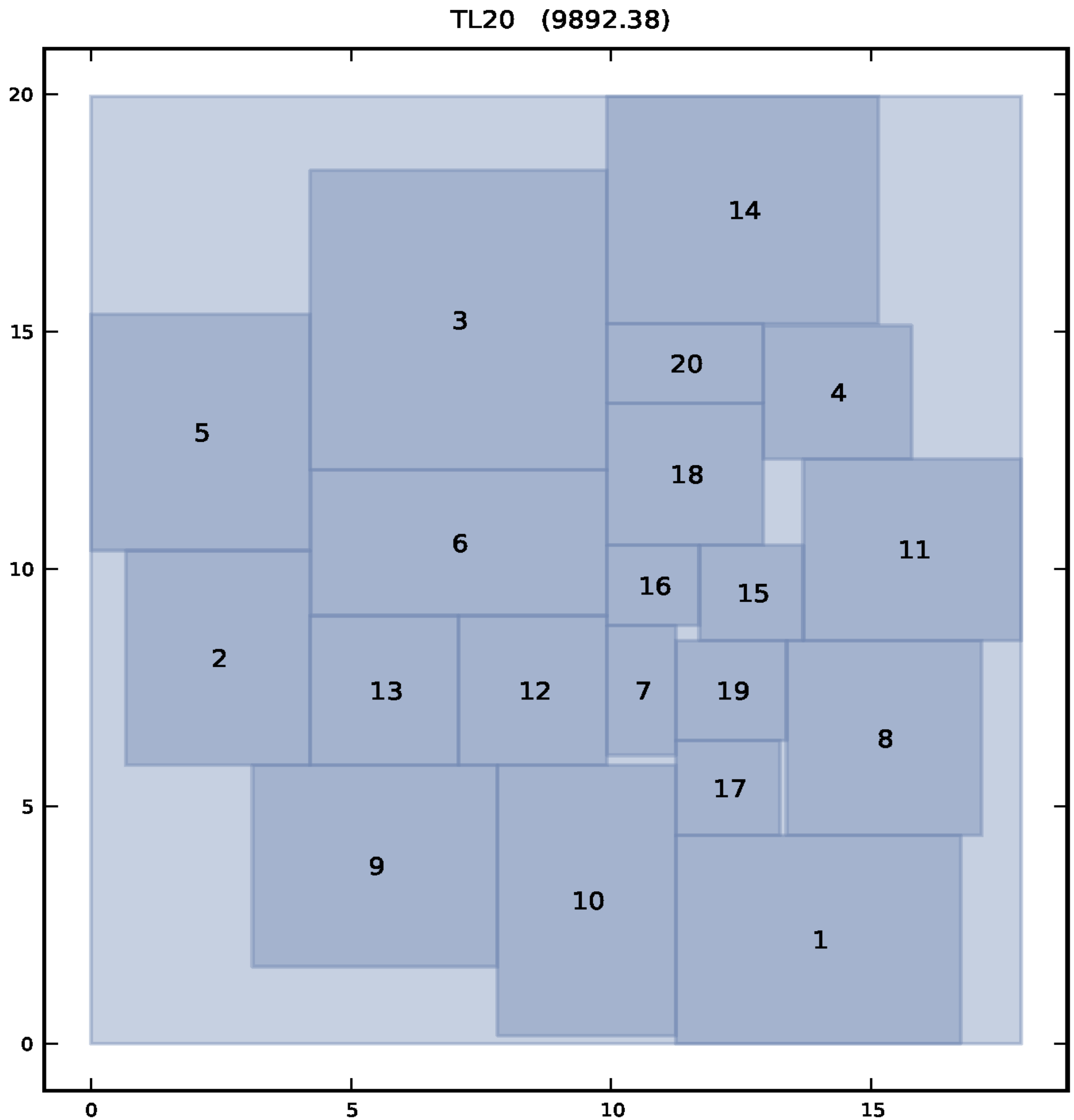
Previous best known
Solution: 559.0
HA-C (Tam and Li, 1991)



TL12

New best known
Solution: 2920.5

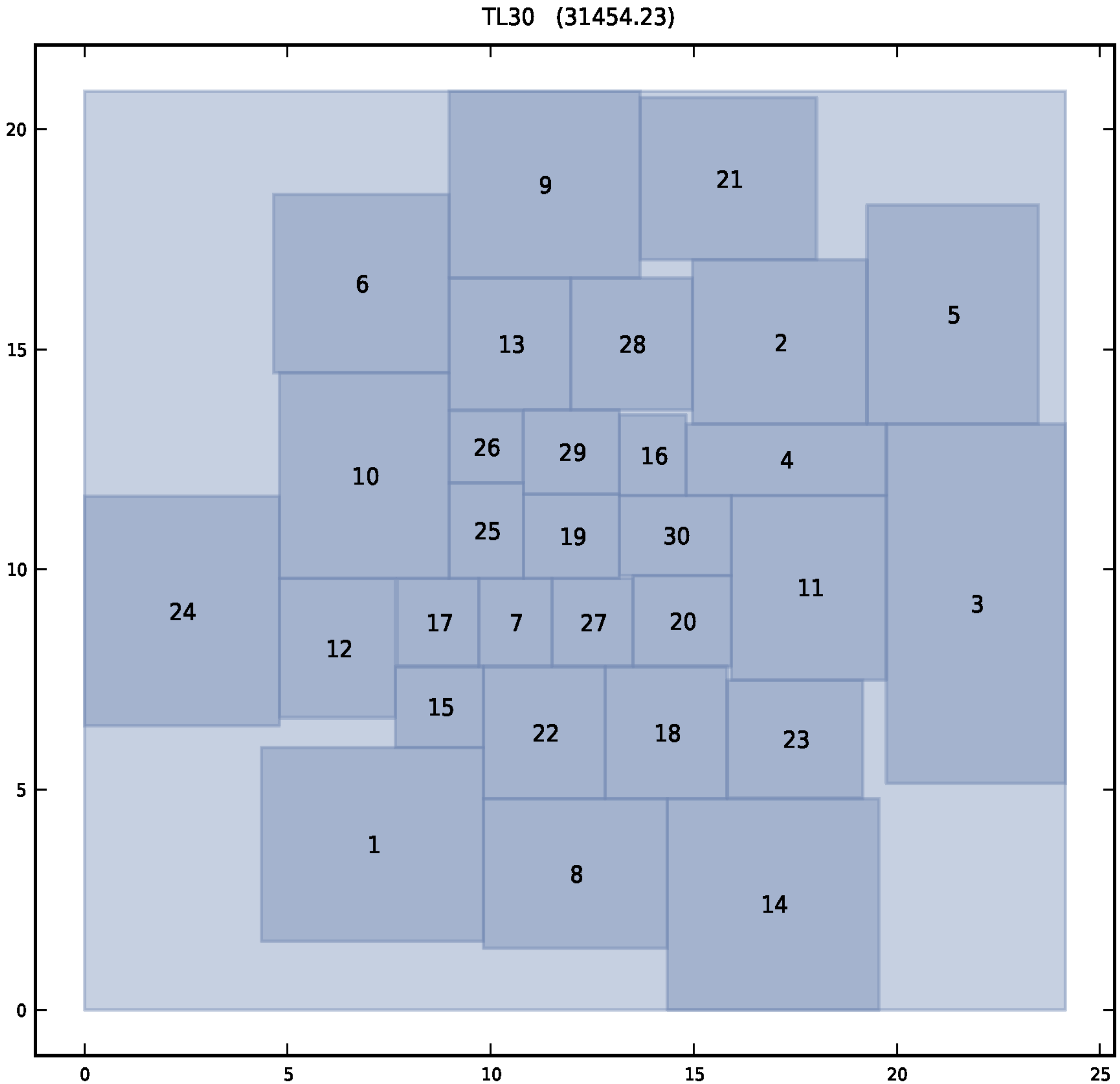
Previous best known
Solution: 3054.2
TSaST (Scholtz et al., 2009)



TL20

New best known Solution: 9892.4

Previous best known Solution: 13190.0 GA-TSG (Schnecke and Vornberger, 1997)



TL30

New best known

Solution: 31454.2

Previous best known

Solution: 33721.5

TSaST (Scholtz et al., 2009)

Concluding remarks

- Reviewed BRKGA framework
- Applied framework to unequal area facility location
  - Presented unconstrained case in this talk
  - Constrained case is presented in paper
- All decoders were simple heuristics
- BRKGA "learned" how to "operate" the heuristics
- In all cases, several new best known solutions were produced for both constrained & unconstrained cases

Thanks!

These slides and all of the papers cited in this talk can be downloaded from my homepage:

http://mauricio.resende.info