A genetic algorithm with random-keys for node placement in path-disjoint network monitoring

DIMACS/DyDAn Workshop on Internet Tomography
DIMACS ~ Rutgers University
May 14-16, 2008.

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

mgcr@research.att.com

DIMACS

Center for Discrete Mathematics & Theoretical Computer Science
Founded as a National Science Foundation Science and Technology Center

Joint work with:

L. Breslau, I. Diakonikolas, N. Duffield, Y. Gu, M. Hajiaghayi, D.S. Johnson, H, Karloff, M.G.C.R., S. Sen, and D. Towsley, “Optimal Node Placement for Path Disjoint Network Monitoring,” AT&T Labs Research Technical Report, November 18, 2007.

Summary

- Minimum monitoring set (MMS) problem
- Algorithms for MMS
- Greedy algorithm
- Genetic algorithm
- Computational experiments
- Concluding remarks

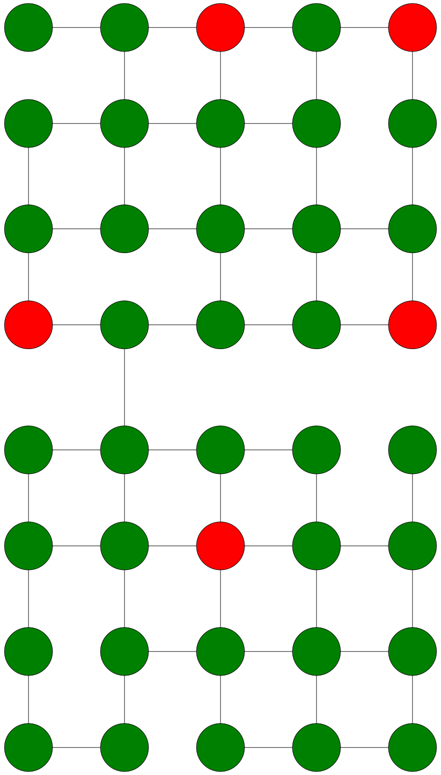
Minimum monitoring set problem

We wish to perform the tomographic inference of hop-on and hop-off performance for each provider edge router:

Deploy a set of N measurement hosts $\{M_1, M_2, ..., M_N\}$ such that for each provider edge router A, there are two measurement hosts M_i and M_j such that the physical paths (A, M_i) and (A, M_j) are disjoint.

One objective is to minimize N.

paths are given and are fixed



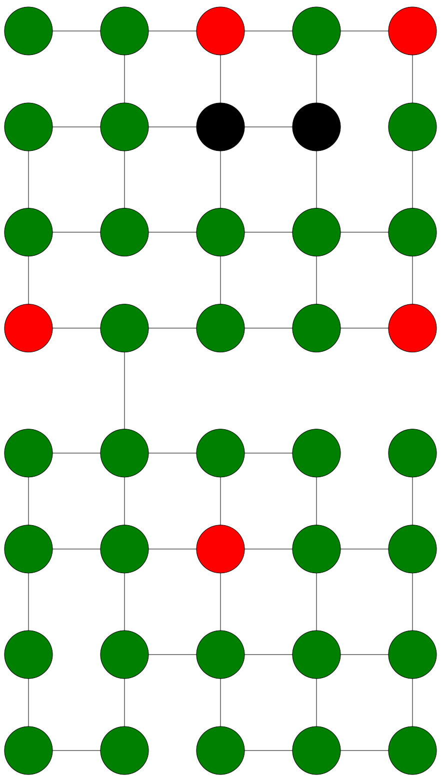
at&t Your world. Delivered.

Minimum monitoring set problem

We wish to perform the tomographic inference of hop-on and hop-off performance for each provider edge router:

- Deploy a set of N measurement hosts $\{M_1, M_2, \ldots, M_N\}$ such that for each provider edge router A, there are two measurement hosts M_i and M_j such that the physical paths (A, M_i) and (A, M_j) are disjoint.
- One objective is to minimize N.

paths are given and are fixed



at&t Your world. Delivered.

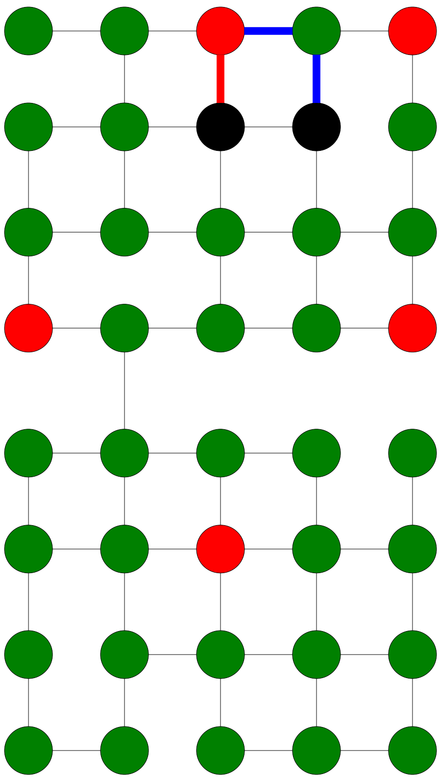
Minimum monitoring set problem

We wish to perform the tomographic inference of hop-on and hop-off performance for each provider edge router:

Deploy a set of N measurement hosts $\{M_1, M_2, \ldots, M_N\}$ such that for each provider edge router A, there are two measurement hosts M_i and M_j such that the physical paths (A, M_i) and (A, M_j) are disjoint.

One objective is to minimize N.

paths are given and are fixed



at&t Your world. Delivered.

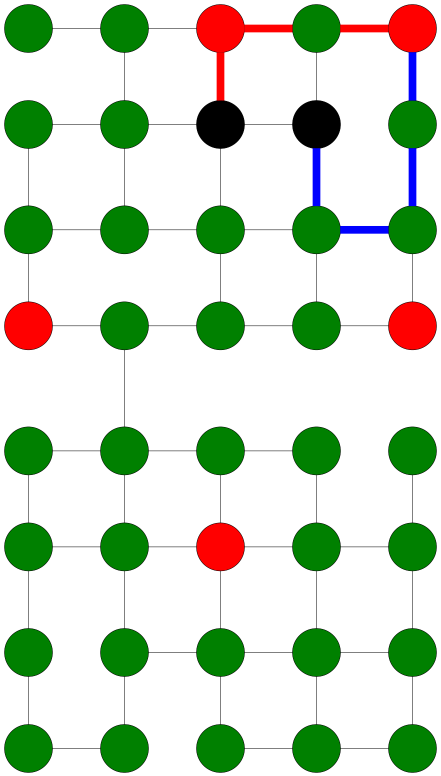
Minimum monitoring set problem

We wish to perform the tomographic inference of hop-on and hop-off performance for each provider edge router:

Deploy a set of N measurement hosts $\{M_1, M_2, \ldots, M_N\}$ such that for each provider edge router A, there are two measurement hosts M_i and M_j such that the physical paths (A, M_i) and (A, M_j) are disjoint.

One objective is to minimize N.

paths are given and are fixed



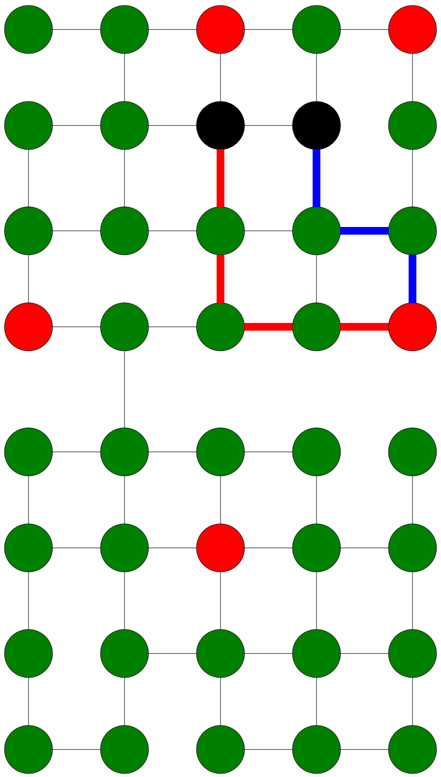
Minimum monitoring set problem

We wish to perform the tomographic inference of hop-on and hop-off performance for each provider edge router:

Deploy a set of N measurement hosts $\{M_1, M_2, \ldots, M_N\}$ such that for each provider edge router A, there are two measurement hosts M_i and M_j such that the physical paths (A, M_i) and (A, M_j) are disjoint.

One objective is to minimize N.

paths are given and are fixed

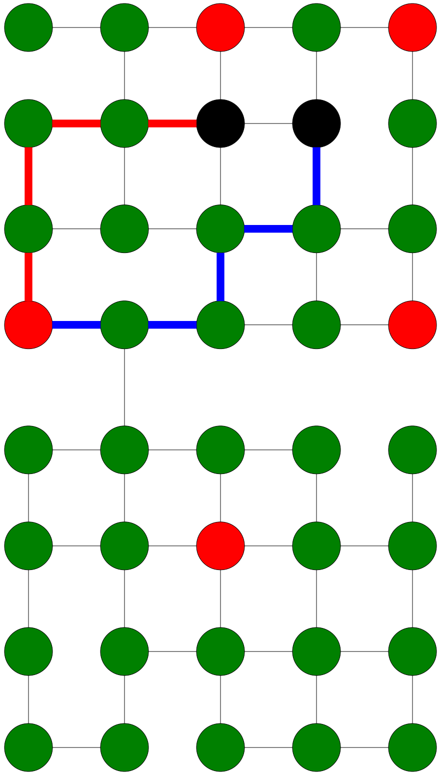


Minimum monitoring set problem

We wish to perform the tomographic inference of hop-on and hop-off performance for each provider edge router:

- Deploy a set of N measurement hosts $\{M_1, M_2, \ldots, M_N\}$ such that for each provider edge router A, there are two measurement hosts M_i and M_j such that the physical paths (A, M_i) and (A, M_j) are disjoint.
- One objective is to minimize N.

paths are given and are fixed



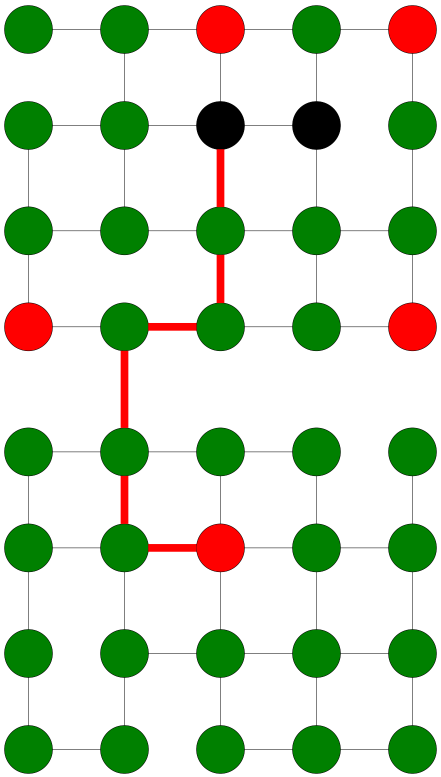
Minimum monitoring set problem

We wish to perform the tomographic inference of hop-on and hop-off performance for each provider edge router:

Deploy a set of N measurement hosts $\{M_1, M_2, \ldots, M_N\}$ such that for each provider edge router A, there are two measurement hosts M_i and M_j such that the physical paths (A, M_i) and (A, M_j) are disjoint.

One objective is to minimize N.

paths are given and are fixed



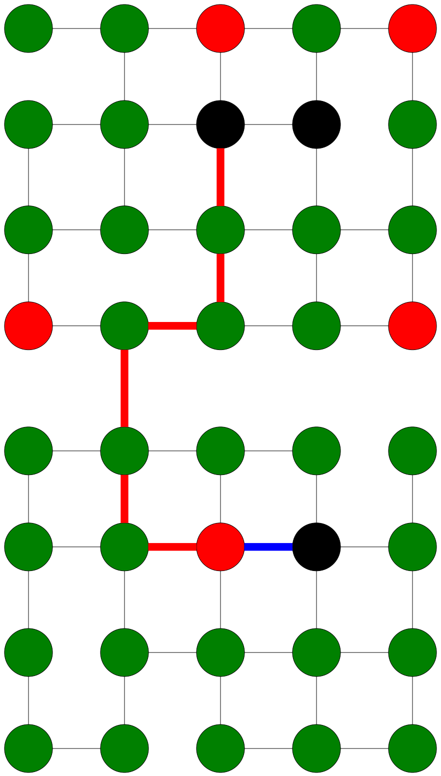
Minimum monitoring set problem

We wish to perform the tomographic inference of hop-on and hop-off performance for each provider edge router:

Deploy a set of N measurement hosts $\{M_1, M_2, ..., M_N\}$ such that for each provider edge router A, there are two measurement hosts M_i and M_j such that the physical paths (A, M_i) and (A, M_j) are disjoint.

One objective is to minimize N.

paths are given and are fixed



at&t
Your world. Delivered.

Set covering with pairs

- Set covering with pairs (SCP) was introduced by Hassin & Segev (2005):
  - GIVEN a ground set X of elements and a set Y of cover items, and for each $x \in X$ a set P_x of pairs of items in Y that cover x. A subset $Y' \subseteq Y$ covers X if for each $x \in X$ one of the pairs in P_x is contained in Y', FIND a minimum-size covering subset.
- SCP is NP-hard and, unless P = NP, is hard to approximate.

Minimum monitoring set problem

- The MMS problem is a special case of SCP. We prove that:
  - Let R(w,u) be the set of all routes from w to u
  - MMS is at least as hard to approximate as SCP, even if:
    - Each set R(w,u) is the set of all shortest paths from w to u;
    - Each set R(w,u) contains only one item, and that is a shortest path from w to u
- However, if we allow arbitrary disjoint paths, then using dynamic programming, the problem can be solved in O(|V|+|E|) time.

at&t
Your world. Delivered.

Algorithms for MMS problem

- Exact integer programming model
- Dynamic programming for arbitrary paths variant
- Greedy heuristic
- Genetic algorithm (heuristic)
- Double hitting set heuristic (HH)
- Lower bound derived from HH

Algorithms for MMS problem

- Exact integer programming model
- Dynamic programming for arbitrary paths variant
- Greedy heuristic
- Genetic algorithm (heuristic)
- Double hitting set heuristic (HH)
- Lower bound derived from HH

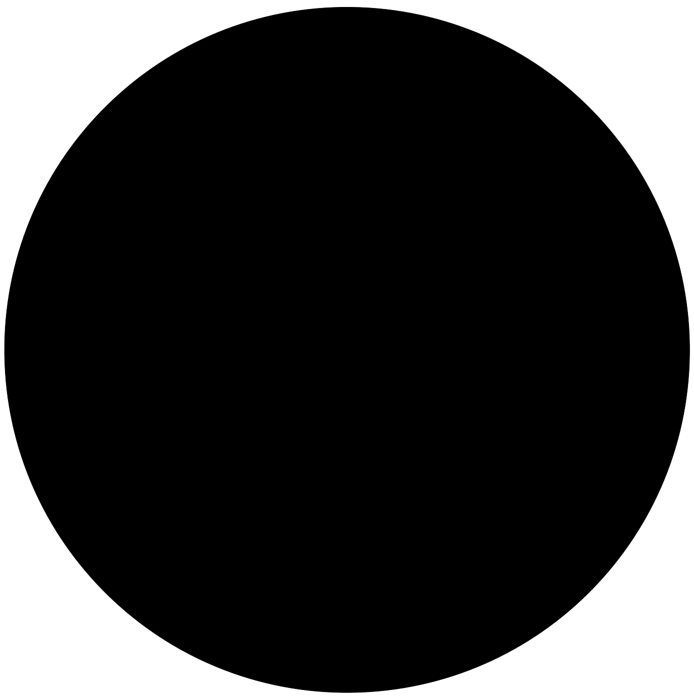
Greedy algorithm for MMS problem

- initialize partial cover $S = \{ \}$
- while S is not a cover do:
  - find $m \in M \setminus S$ such that $S \cup \{m\}$ covers a maximum number of additional branch nodes (break ties by vertex index) and set $S = S \cup \{m\}$
  - if no $m \in M \setminus S$ yields an increase in coverage, then choose a pair $\{m_1,m_2\} \in M \setminus S$ that yields a maximum increase in coverage and set $S = S \cup \{m_1\} \cup \{m_2\}$
  - if no pair exists, then the problem is infeasible

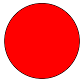
at&t
Your world. Delivered.

Genetic algorithms

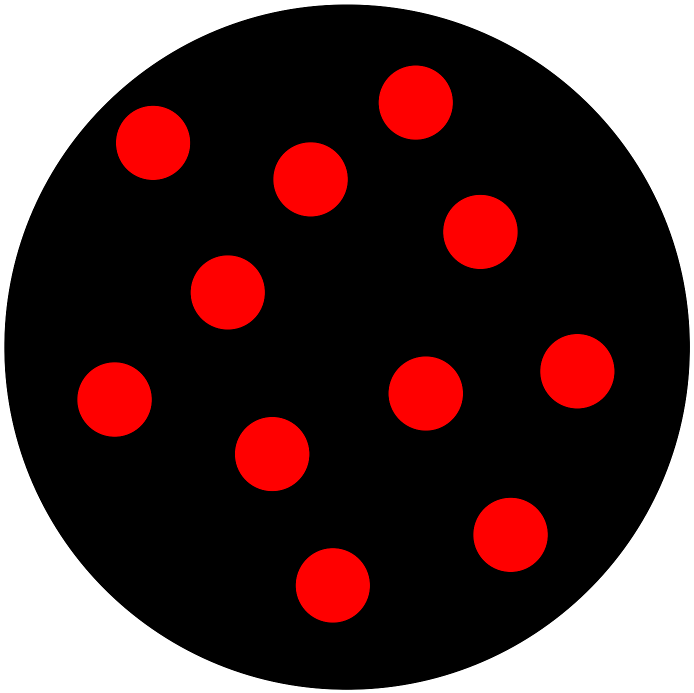
Genetic algorithms Holland (1975)



Adaptive methods that are used to solve search and optimization problems.

Individual: solution 

Genetic algorithms

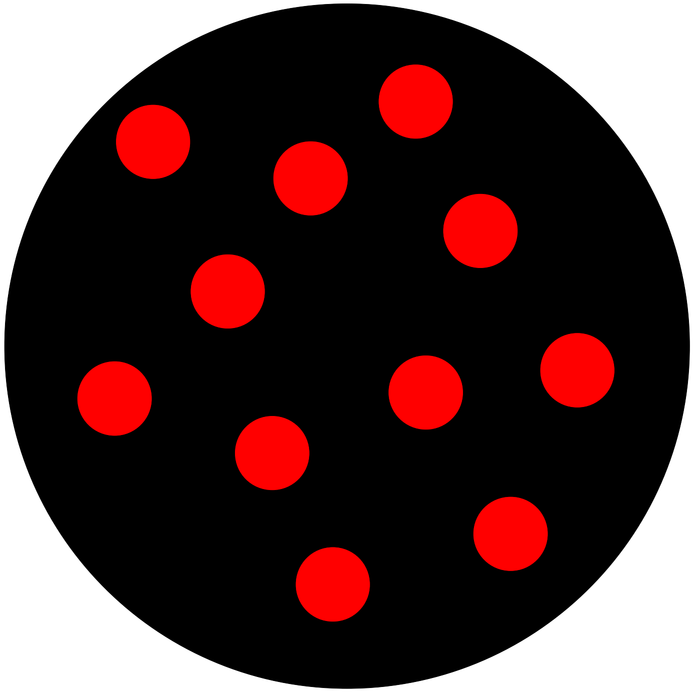


Individual: solution

Population: set of fixed number of individuals

at&t
Your world. Delivered.

Genetic algorithms

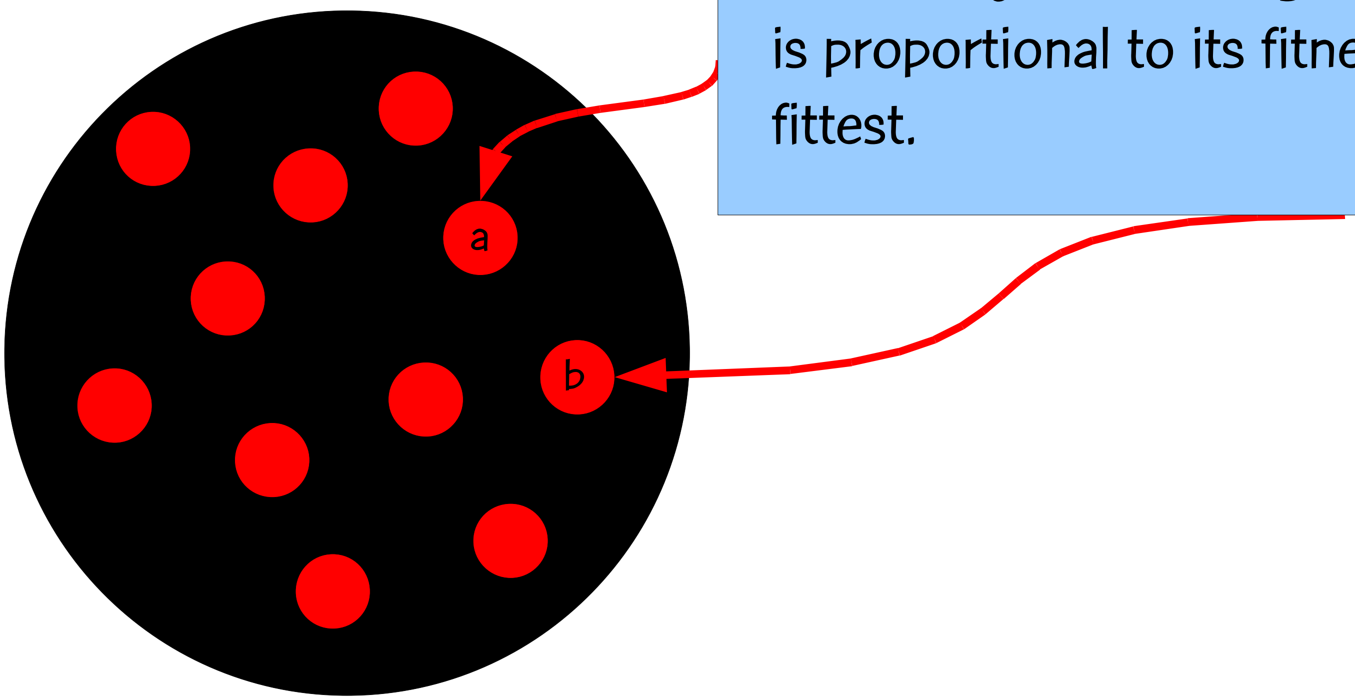


Genetic algorithms evolve population applying the principle of survival of the fittest.

A series of generations are produced by the algorithm. The most fit individual of last generation is the solution.

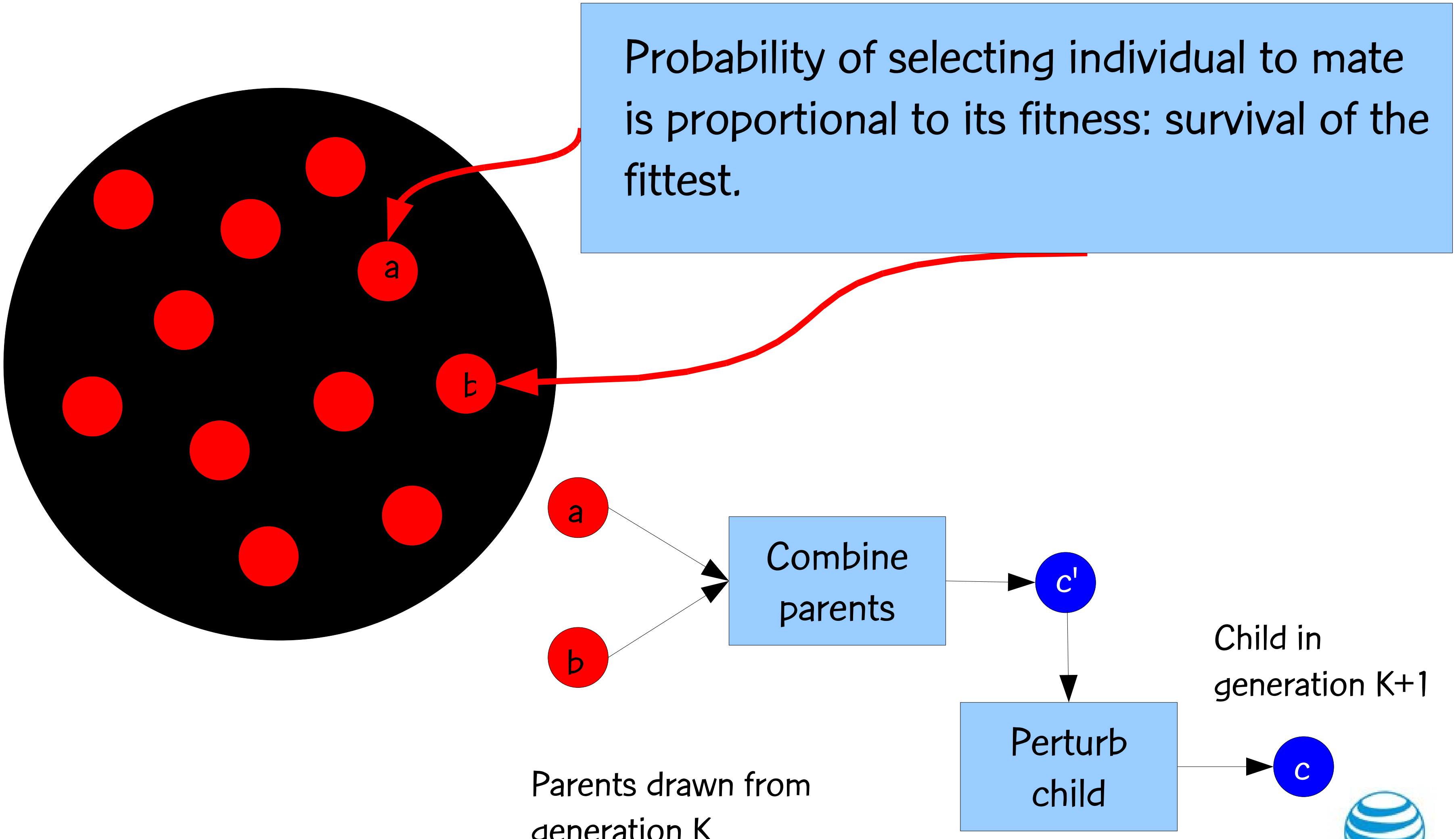
Individuals from one generation are combined to produce offspring that make up next generation.

Genetic algorithms

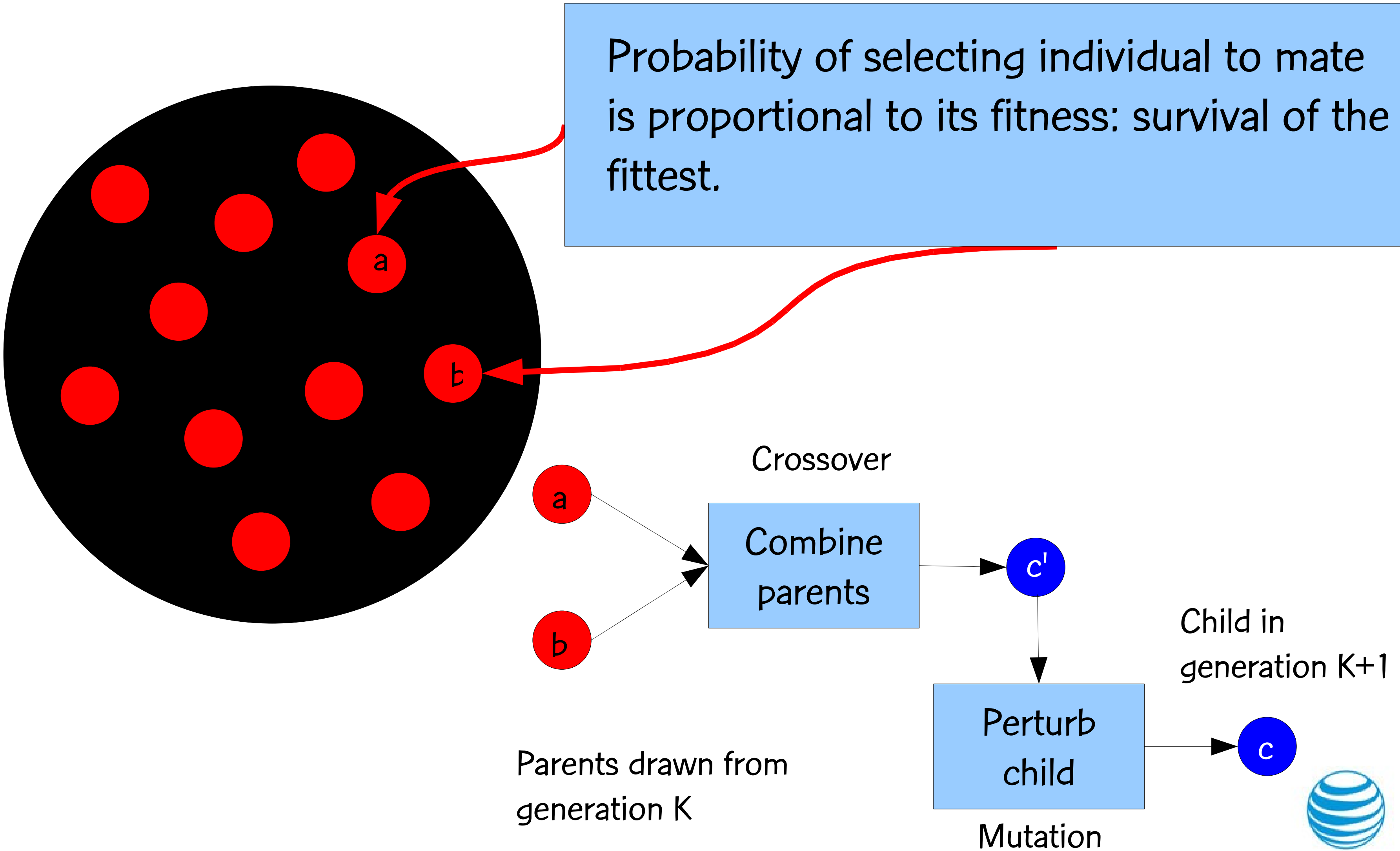


Probability of selecting individual to mate is proportional to its fitness: survival of the fittest.

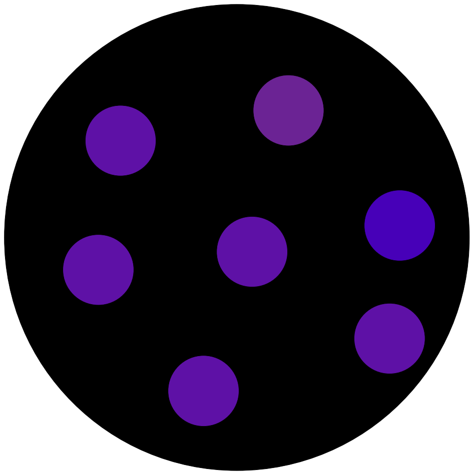
Genetic algorithms



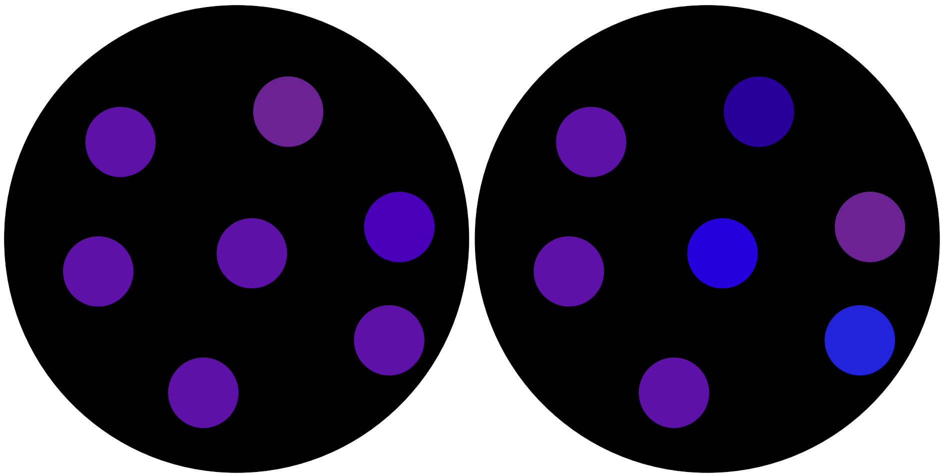
Genetic algorithms



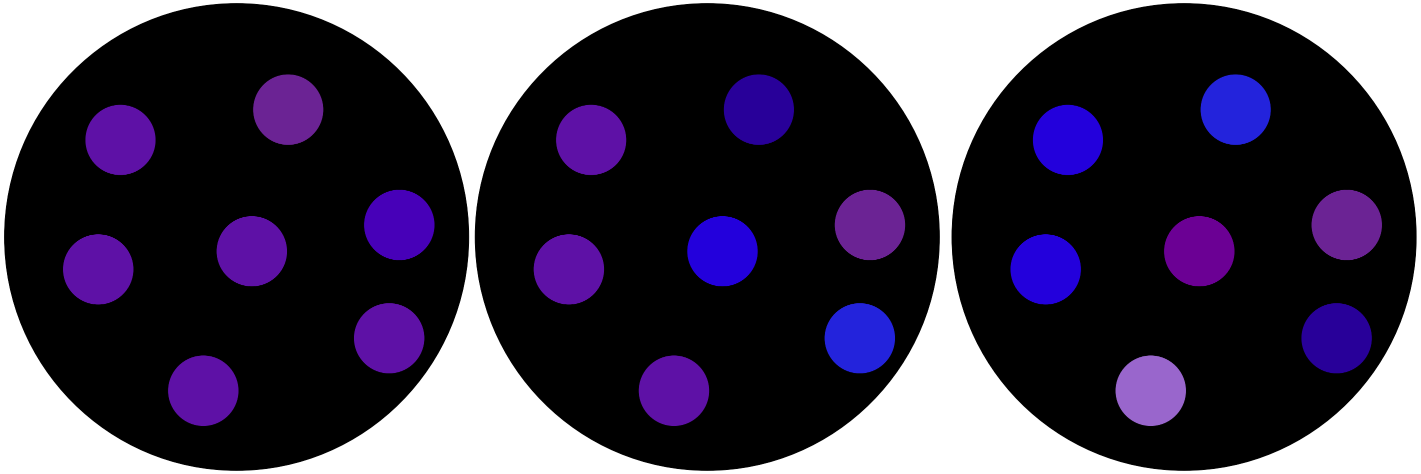
Evolution of solutions



Evolution of solutions

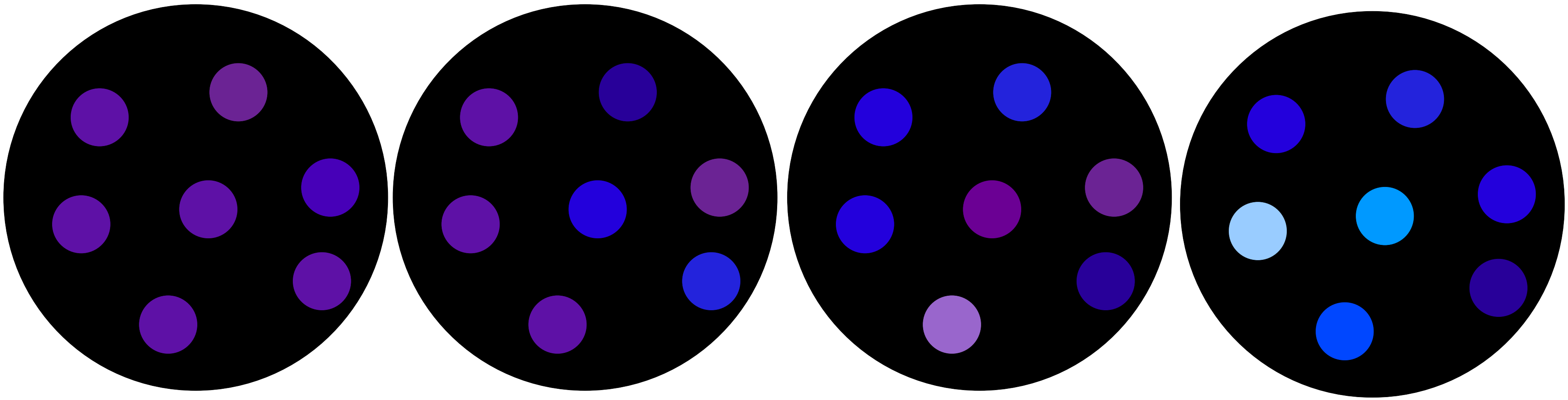


Evolution of solutions

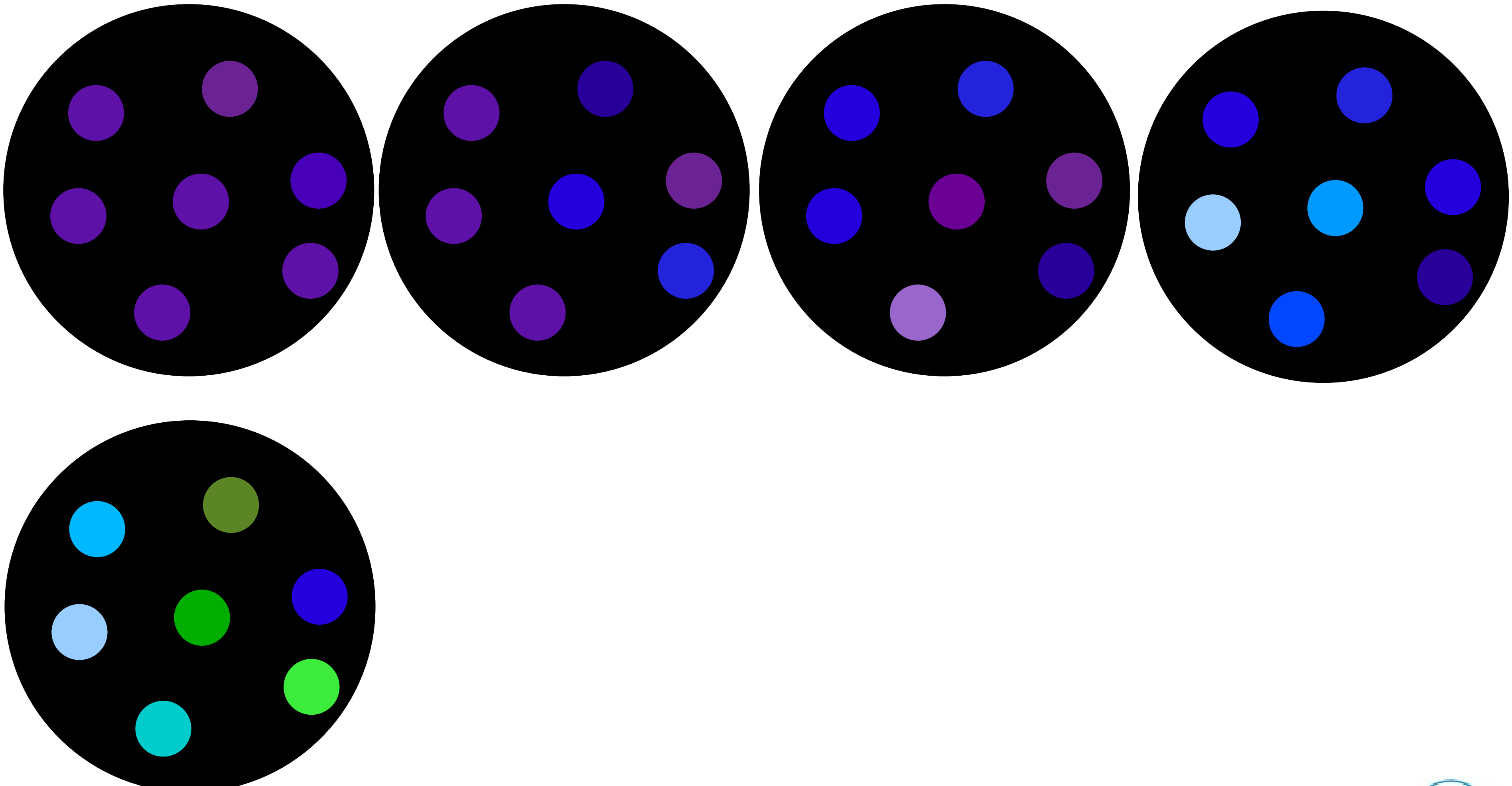


at&t
Your world. Delivered.

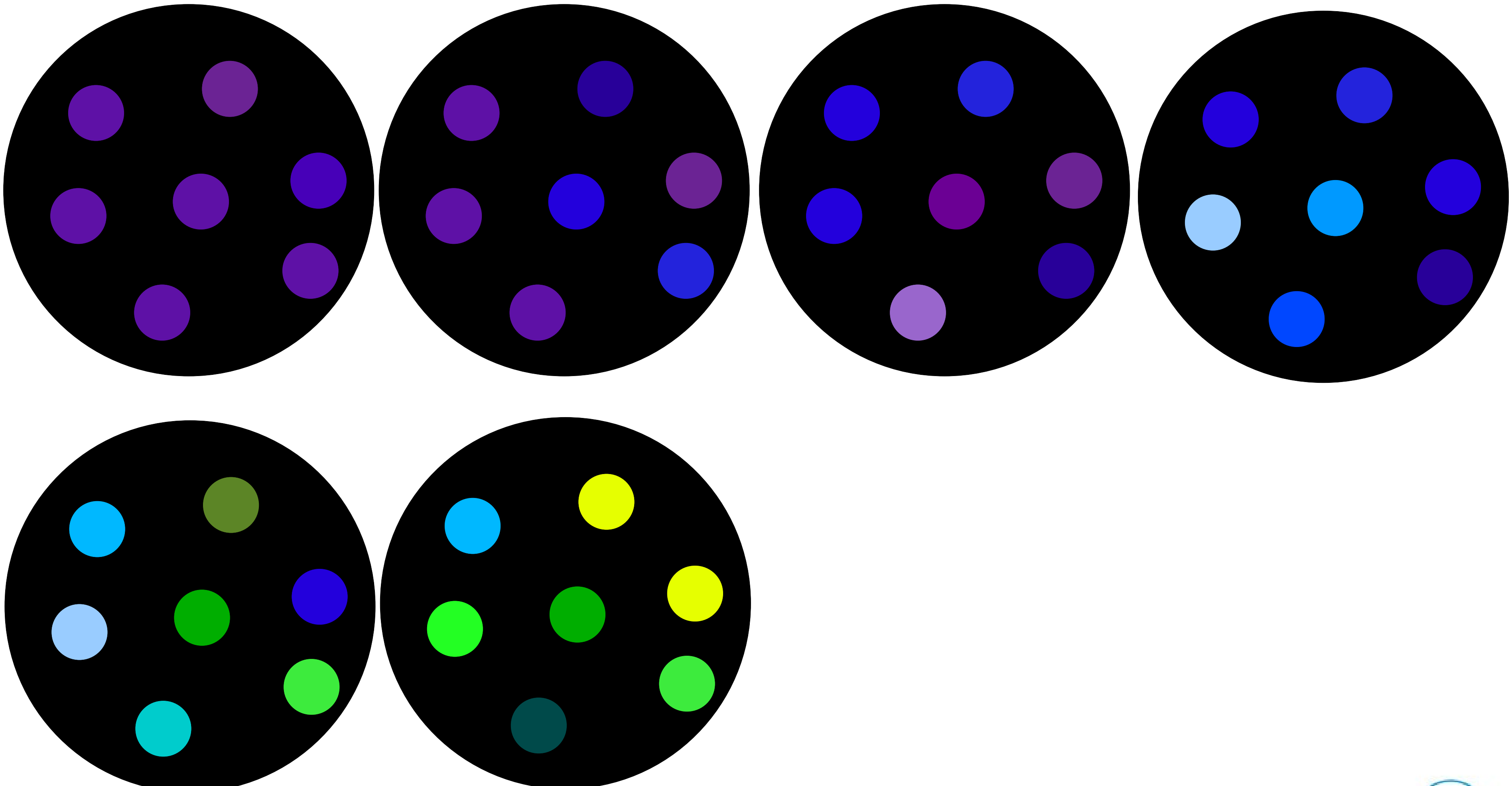
Evolution of solutions



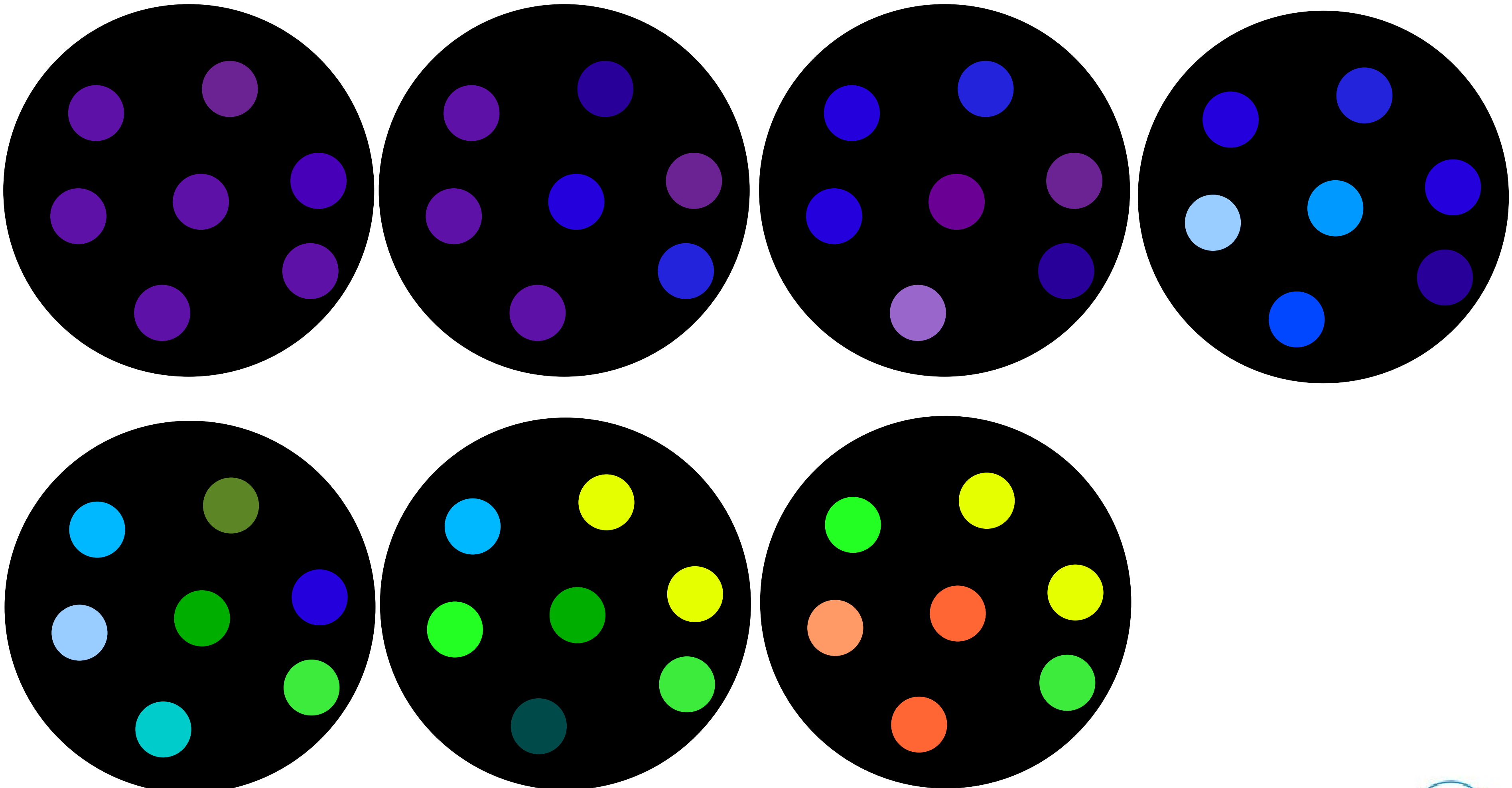
Evolution of solutions



Evolution of solutions

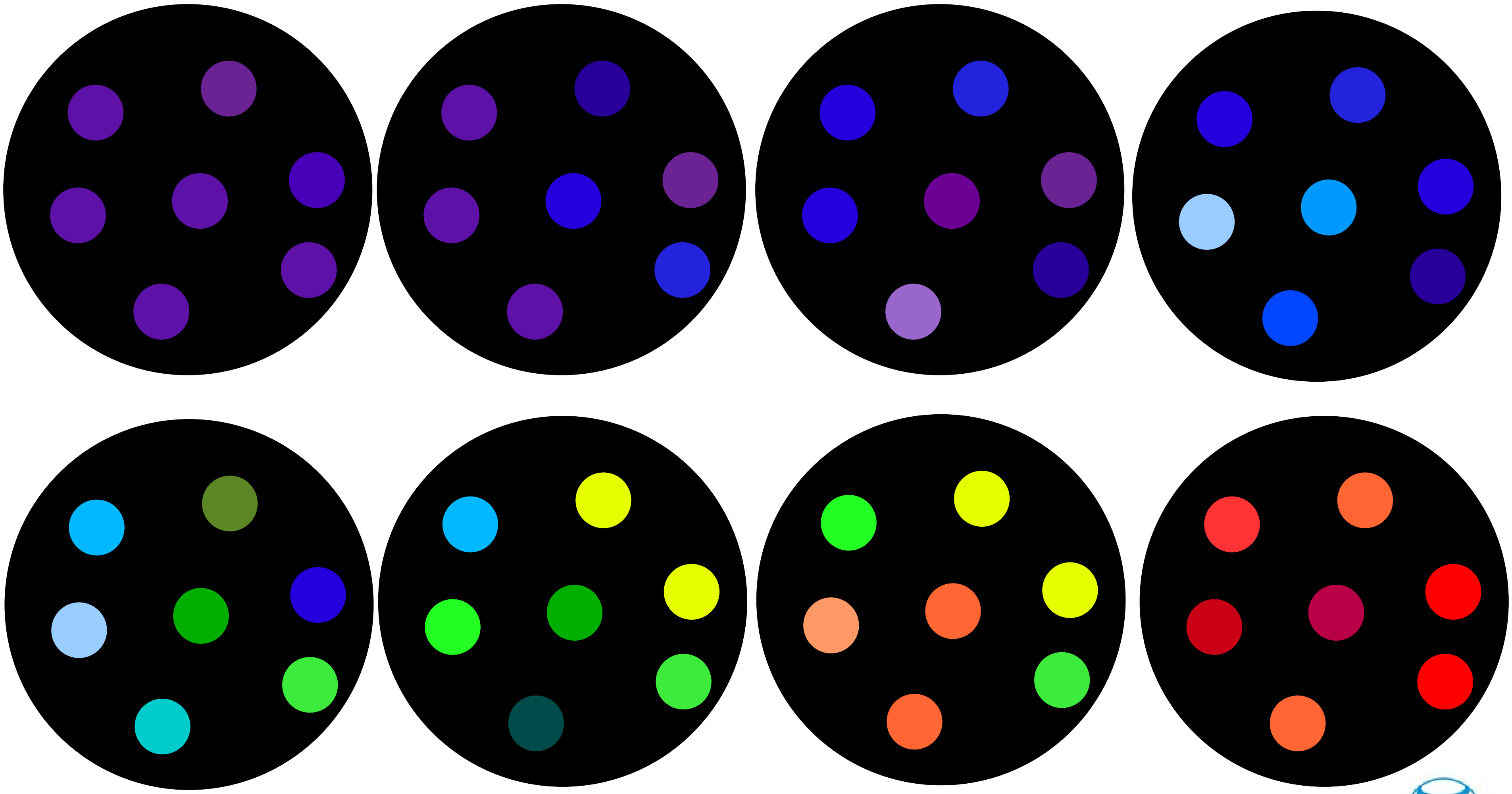


Evolution of solutions



at&t
Your world. Delivered.

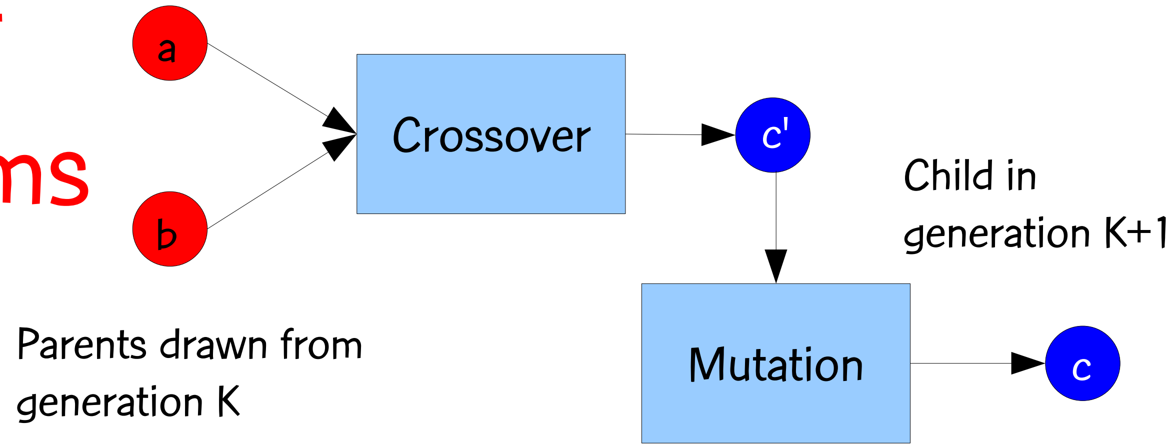
Evolution of solutions



GA lingo

- Population: set of individuals (solutions)
- Chromosome: string (encodes a solution)
- Gene: feature, character, detector (chromosomes are strings of genes)
- Allele: feature value
- Crossover: combination (mating) of two "parent" solutions to produce a "child" solution
- Mutation: perturbation of "child" solution

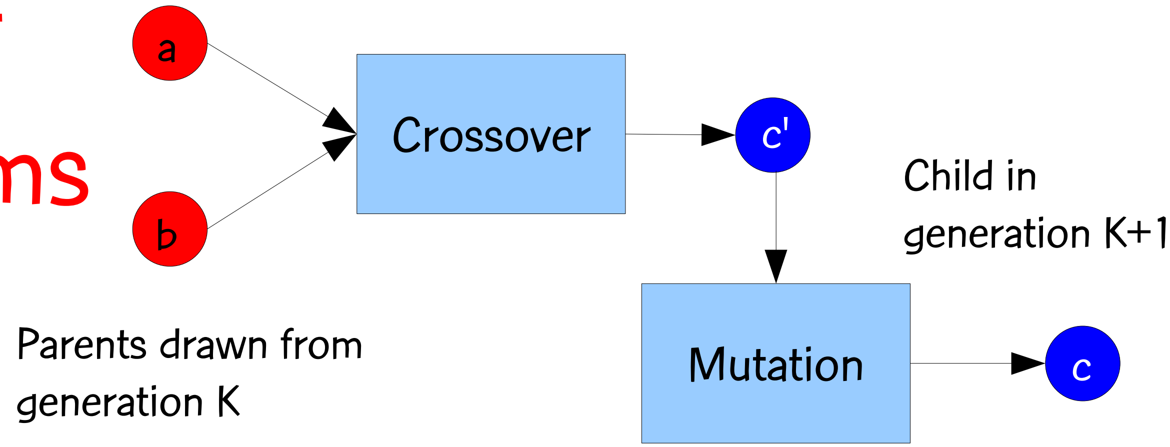
A drawback of genetic algorithms



Given (feasible) parents “a” and “b” in generation K, problem-dependent crossover and mutation operators are needed to guarantee that child “c” in generation K+1 is also feasible.

Therefore, there is a need for specialized representations as well as crossover and mutation operators for each problem variation.

A drawback of genetic algorithms



Given (feasible) parents "a" and "b" in generation K, problem-dependent crossover and mutation operators are needed to guarantee that child "c" in generation K+1 is also feasible.

Therefore, there is a need for specialized representations as well as crossover and mutation operators for each problem variation.

A genetic algorithm with problem independent crossover and mutation operators could be more appealing.

Genetic algorithms with random keys

Your world. Delivered.

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)

at&t
Your world. Delivered.

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

S' =	(0.05,	0.19,	0.25,	0.67,	0.89)
	s(4)	s(2)	s(1)	s(3)	s(5)

Sequence: 4 – 2 – 1 – 3 – 5

GAs and random keys

- Introduced by Bean (1994) for sequencing problems.
- 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

- Introduced by Bean (1994) for sequencing problems.
- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passed the allele 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

- Introduced by Bean (1994) for sequencing problems.
- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passed the allele 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 = ( )

at&t
Your world. Delivered.

GAs and random keys

- Introduced by Bean (1994) for sequencing problems.
- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passed the allele 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

- Introduced by Bean (1994) for sequencing problems.
- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passed the allele 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

- Introduced by Bean (1994) for sequencing problems.
- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passed the allele 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

- Introduced by Bean (1994) for sequencing problems.
- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passed the allele 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

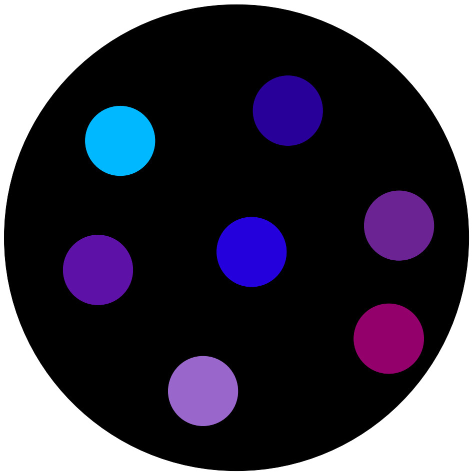
- Introduced by Bean (1994) for sequencing problems.
- Mating is done using parametrized uniform crossover (Spears & DeJong , 1990)
- For each gene, flip a biased coin to choose which parent passed the allele to the child.

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)

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

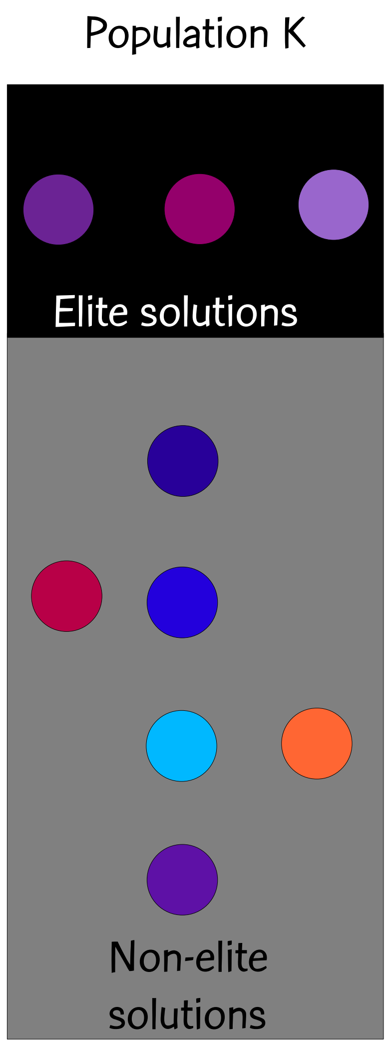
GAs and random keys

- Introduced by Bean (1994) for sequencing problems.
- Initial population is made up of P chromosomes, each with N genes, each having a value (allele) generated uniformly at random in the interval [0,1].



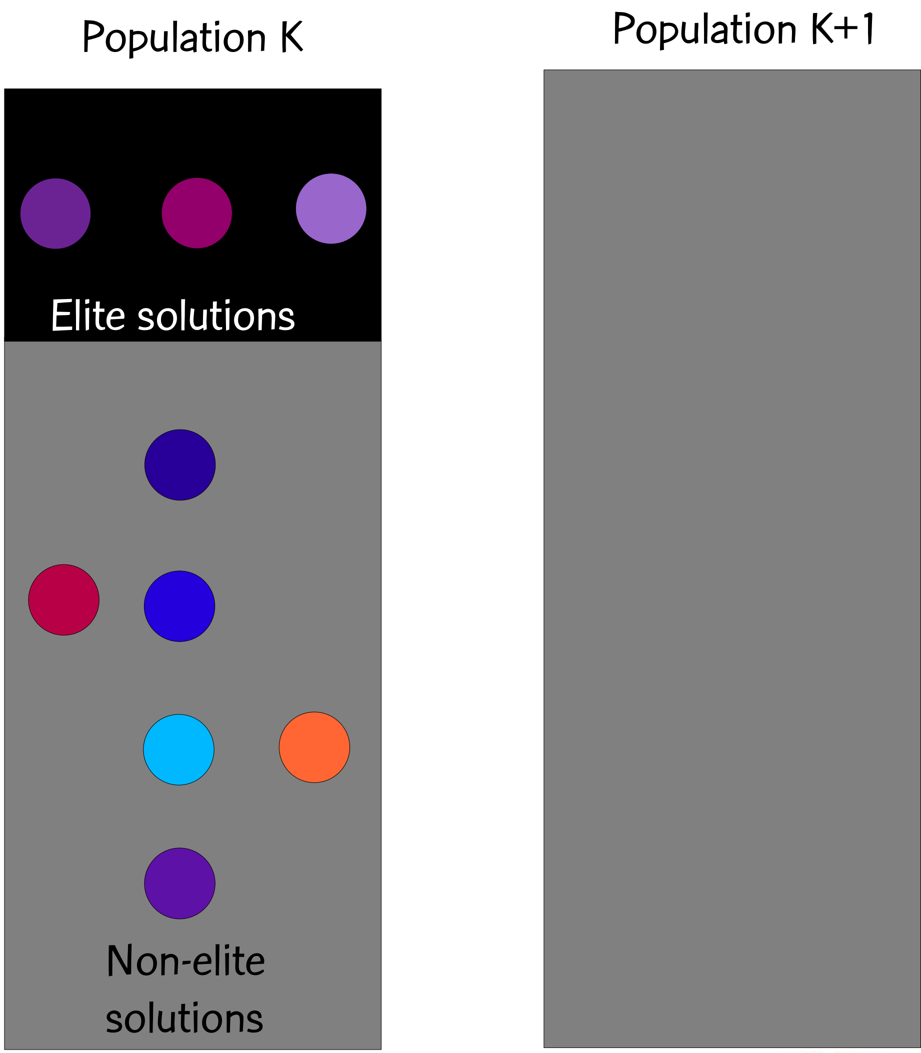
GAs and random keys

- Introduced by Bean (1994) for sequencing problems.
- At the K-th generation, compute the cost of each solution and partition the solutions into two sets: elite solutions, non-elite solutions. Elite set should be smaller of the two sets and contain best solutions.



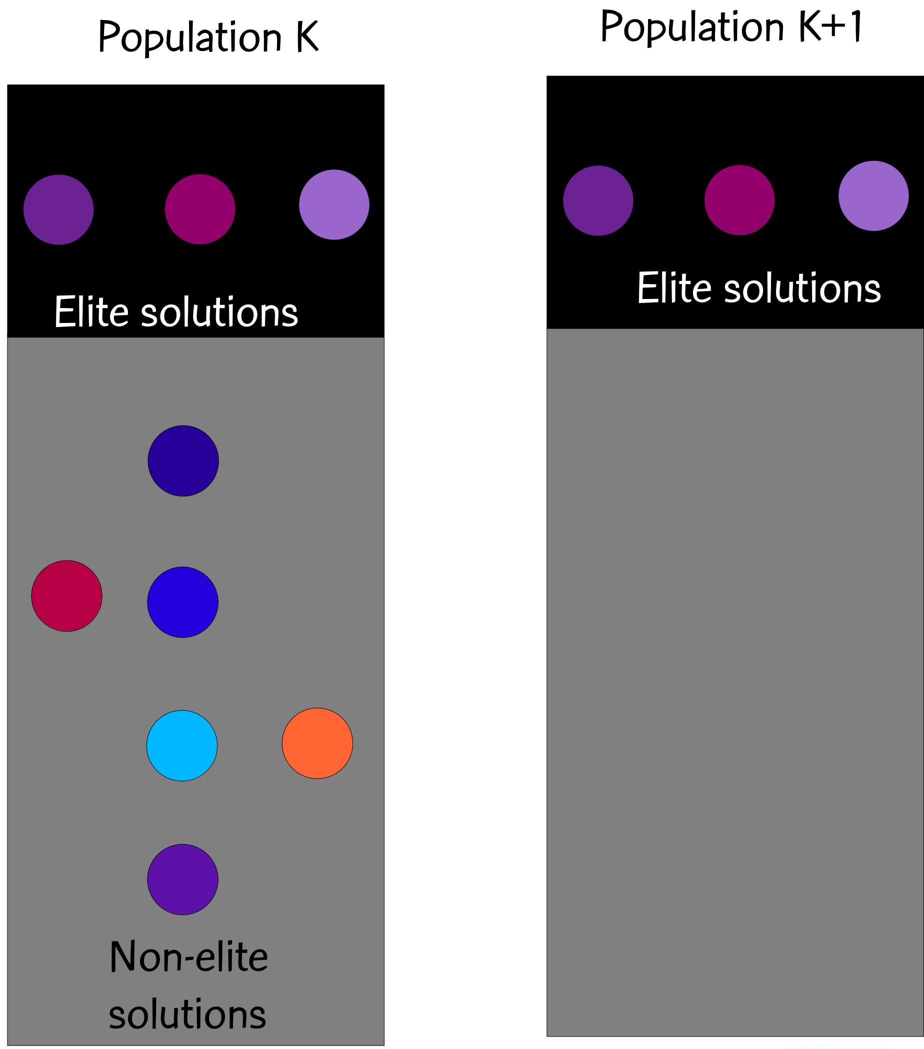
GAs and random keys

- Introduced by Bean (1994) for sequencing problems.
- Evolutionary dynamics



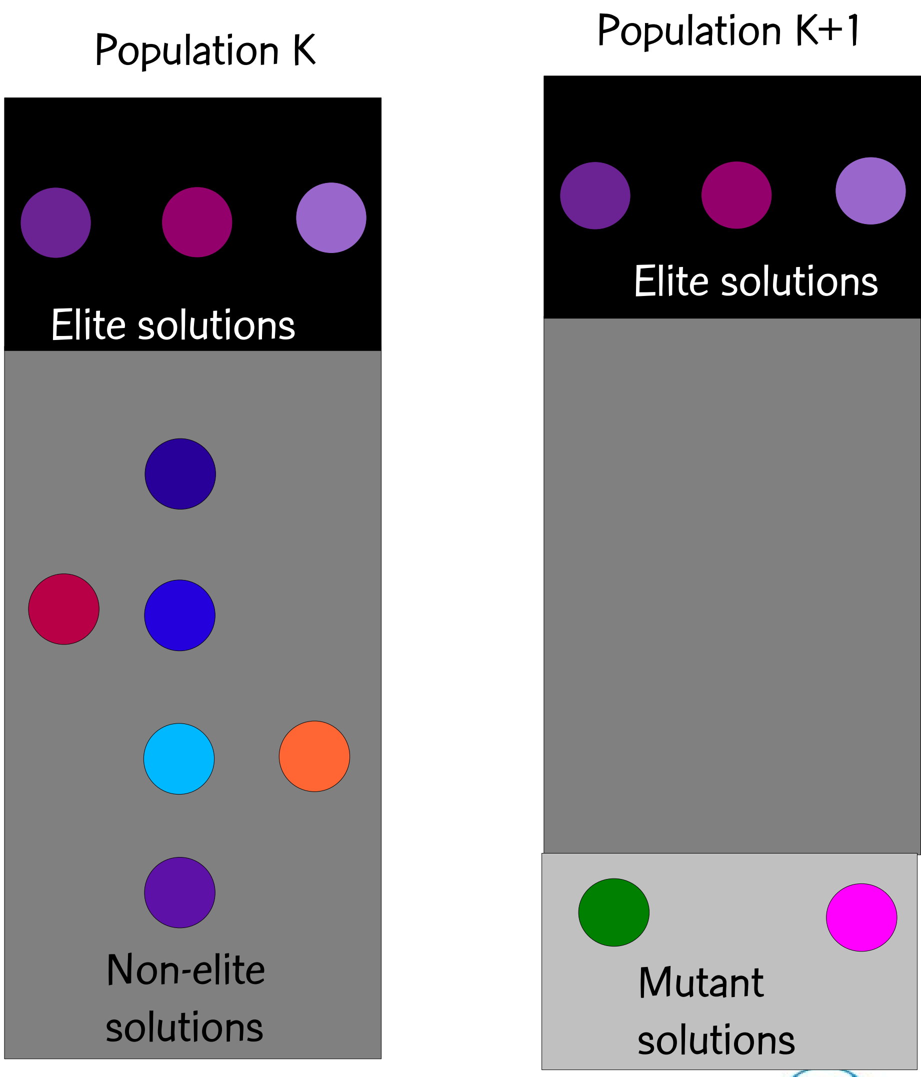
GAs and random keys

- Introduced by Bean (1994) for sequencing problems.
- Evolutionary dynamics
  - Copy elite solutions from population K to population K+1



GAs and random keys

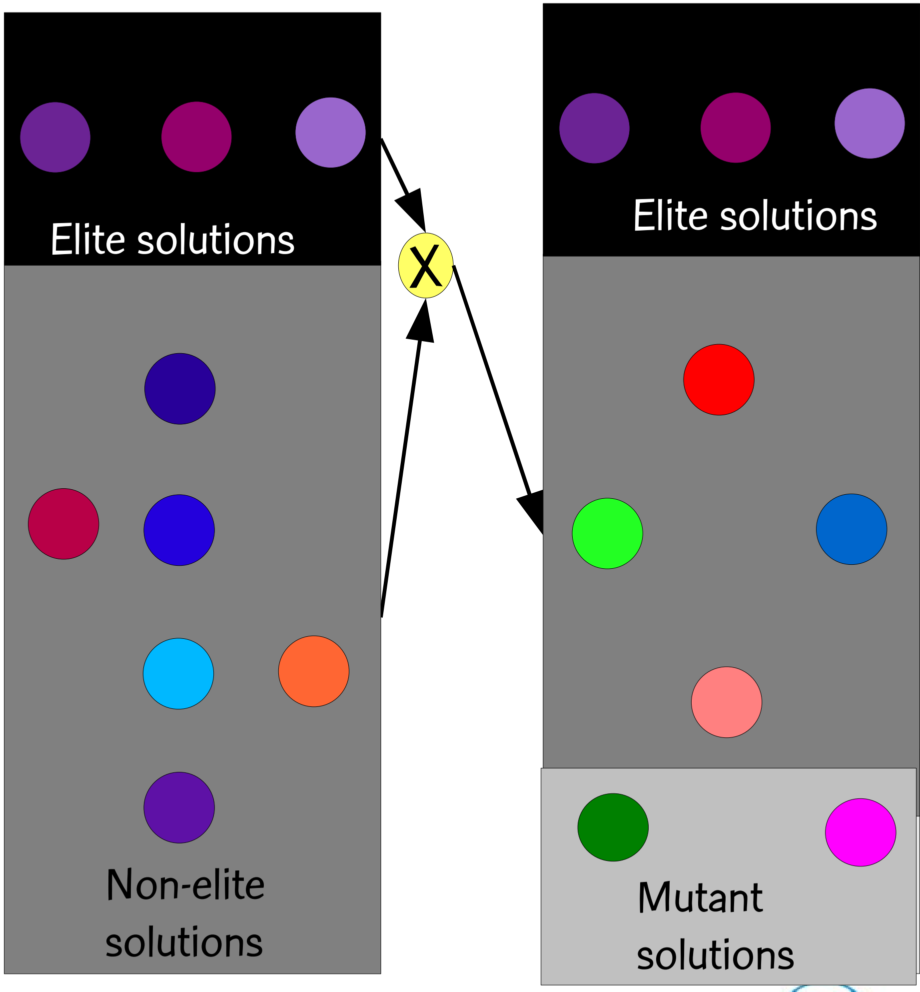
- Introduced by Bean (1994) for sequencing problems.
- Evolutionary dynamics
  - Copy elite solutions from population K to population K+1
  - Add R random solutions (mutants) to population K+1



at&t
Your world. Delivered.

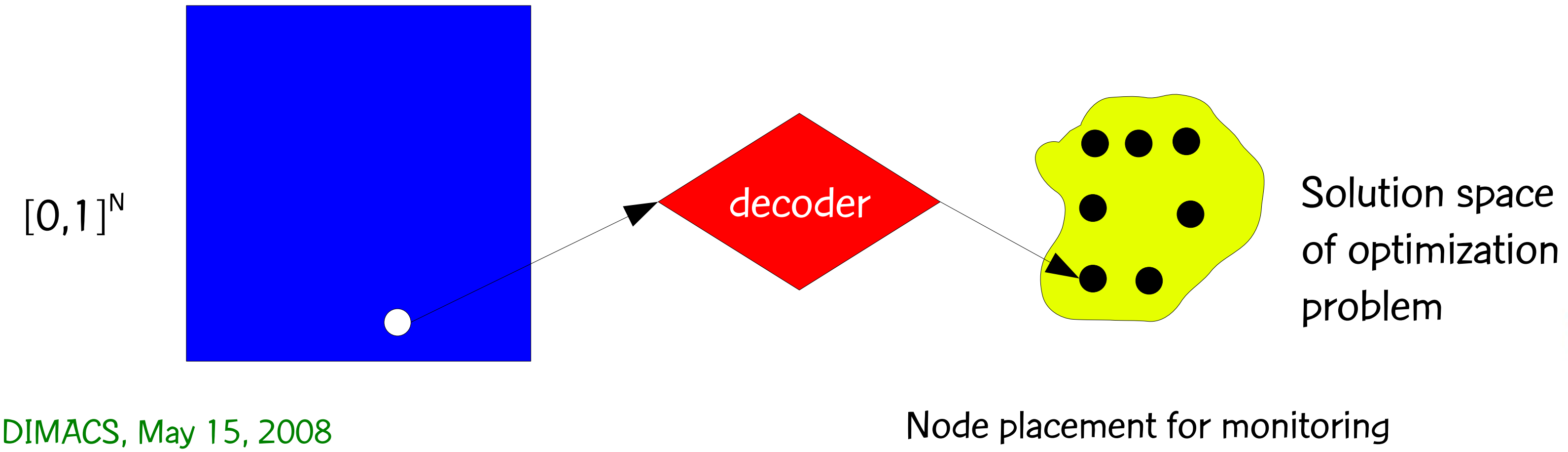
GAs and random keys

- Introduced by Bean (1994) for sequencing problems.
- Evolutionary dynamics
  - Copy elite solutions from population K to population K+1
  - Add R random solutions (mutants) to population K+1
  - While K+1-th population < P
    - Mate elite solution with non elite to produce child in population K+1. Mates are chosen at random.



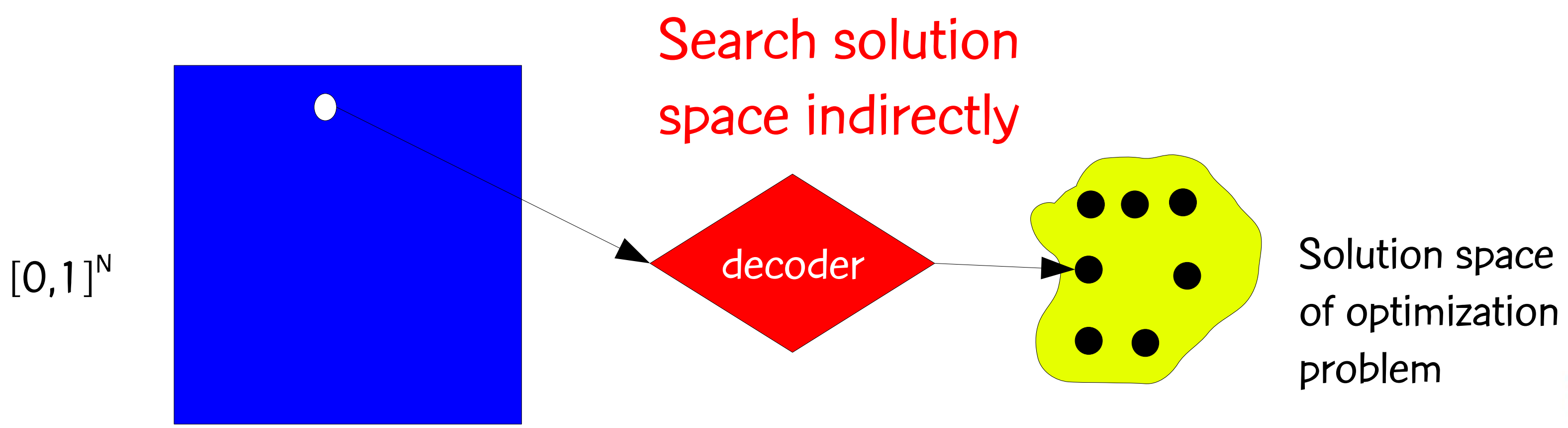
Decoders

- A decoder is a deterministic algorithm that takes as input a random-key vector and returns a feasible solution of the optimization problem and its cost.
- Bean (1994) proposed decoders based on sorting the random-key vector to produce a sequence.
- A random-key GA searches the solution space indirectly by searching the space of random keys and using the decoder to evaluate fitness of the random key.



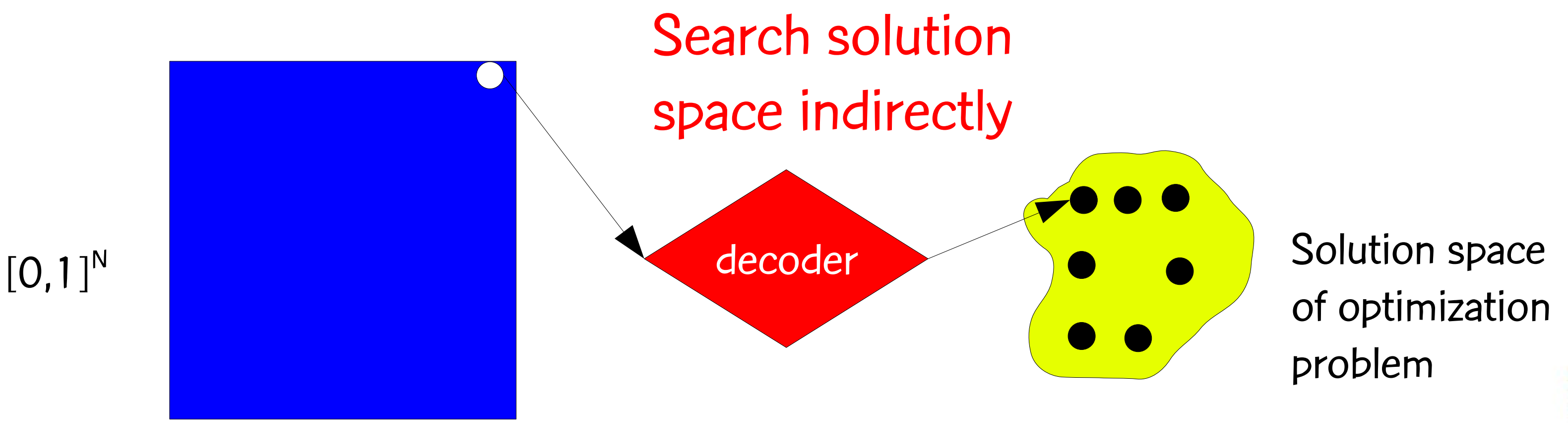
Decoders

- A decoder is a deterministic algorithm that takes as input a random-key vector and returns a feasible solution of the optimization problem and its cost.
- Bean (1994) proposed decoders based on sorting the random-key vector to produce a sequence.
- A random-key GA searches the solution space indirectly by searching the space of random keys and using the decoder to evaluate fitness of the random key.



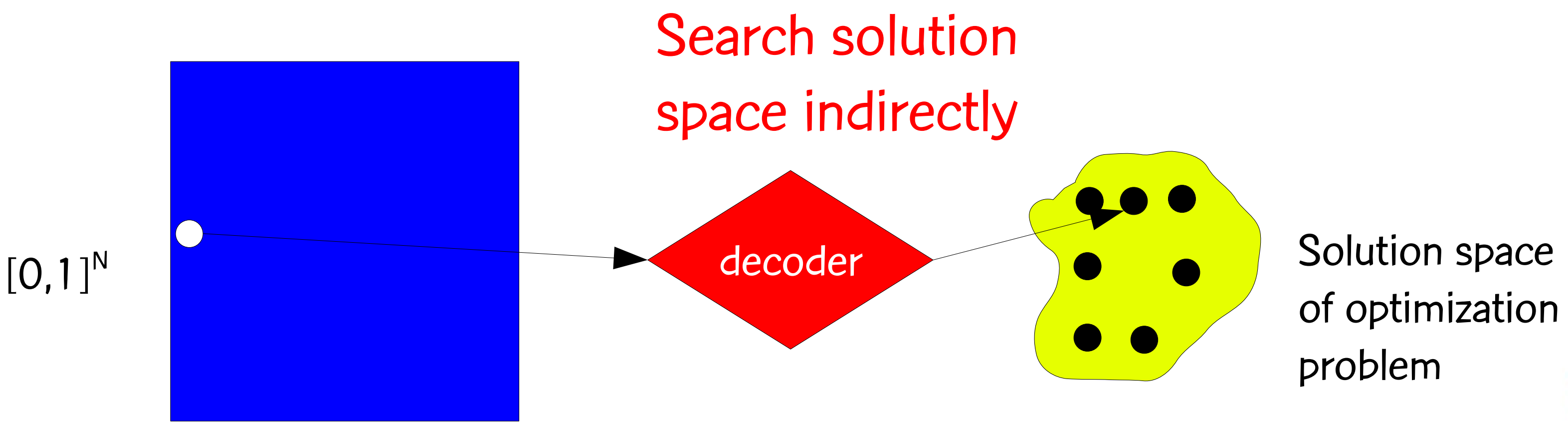
Decoders

- A decoder is a deterministic algorithm that takes as input a random-key vector and returns a feasible solution of the optimization problem and its cost.
- Bean (1994) proposed decoders based on sorting the random-key vector to produce a sequence.
- A random-key GA searches the solution space indirectly by searching the space of random keys and using the decoder to evaluate fitness of the random key.



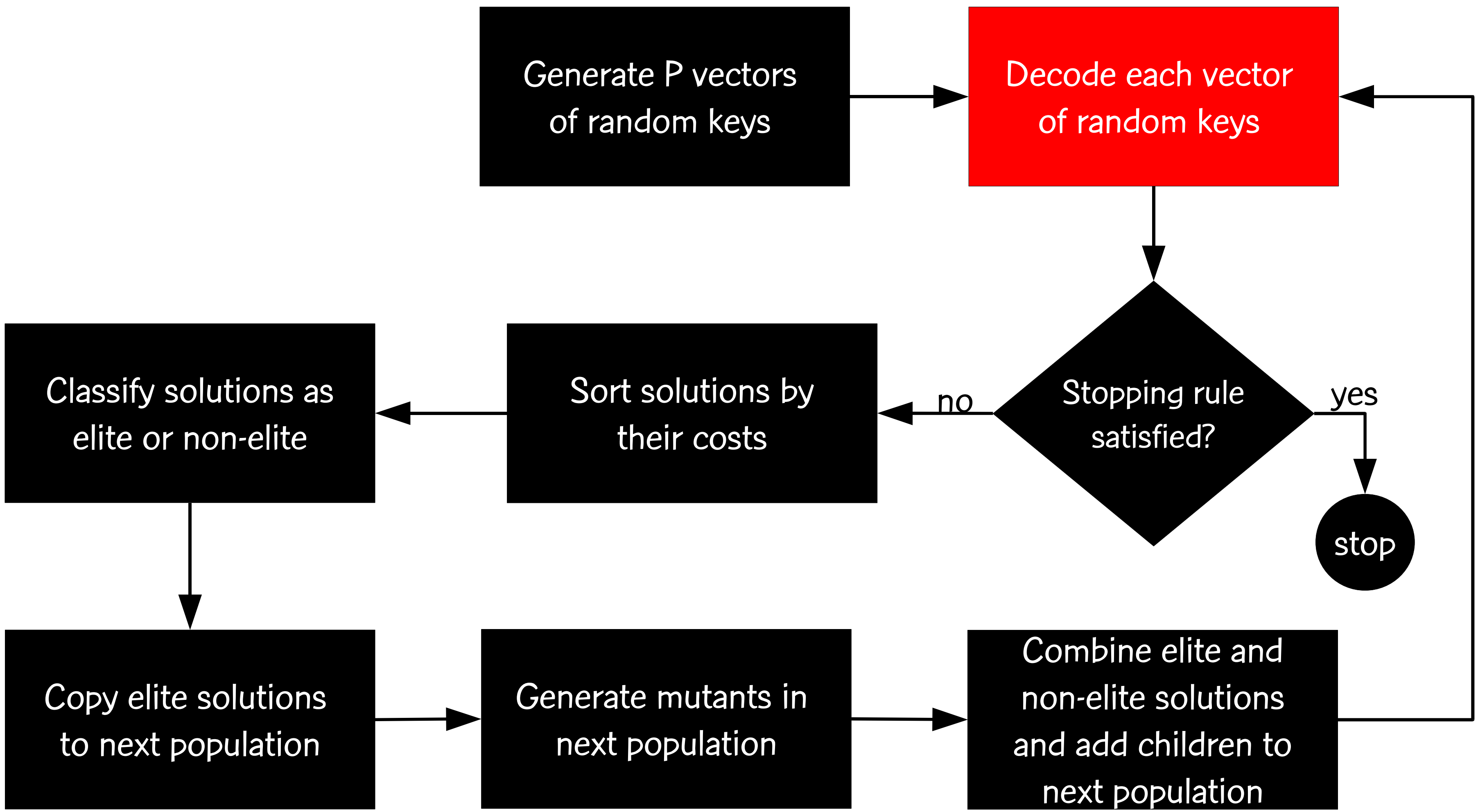
Decoders

- A decoder is a deterministic algorithm that takes as input a random-key vector and returns a feasible solution of the optimization problem and its cost.
- Bean (1994) proposed decoders based on sorting the random-key vector to produce a sequence.
- A random-key GA searches the solution space indirectly by searching the space of random keys and using the decoder to evaluate fitness of the random key.

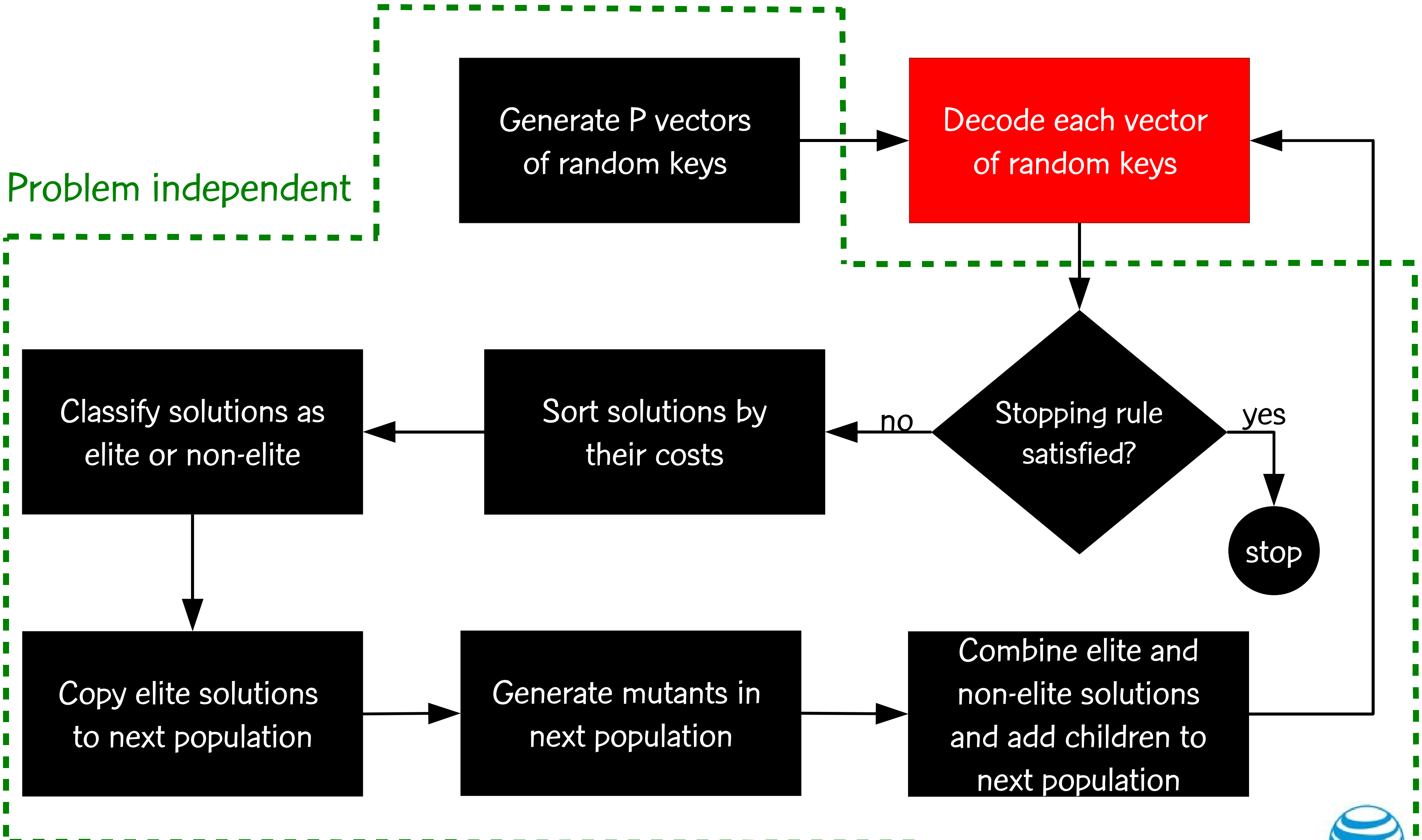


at&t Your world. Delivered.

Framework for random-key genetic algorithms

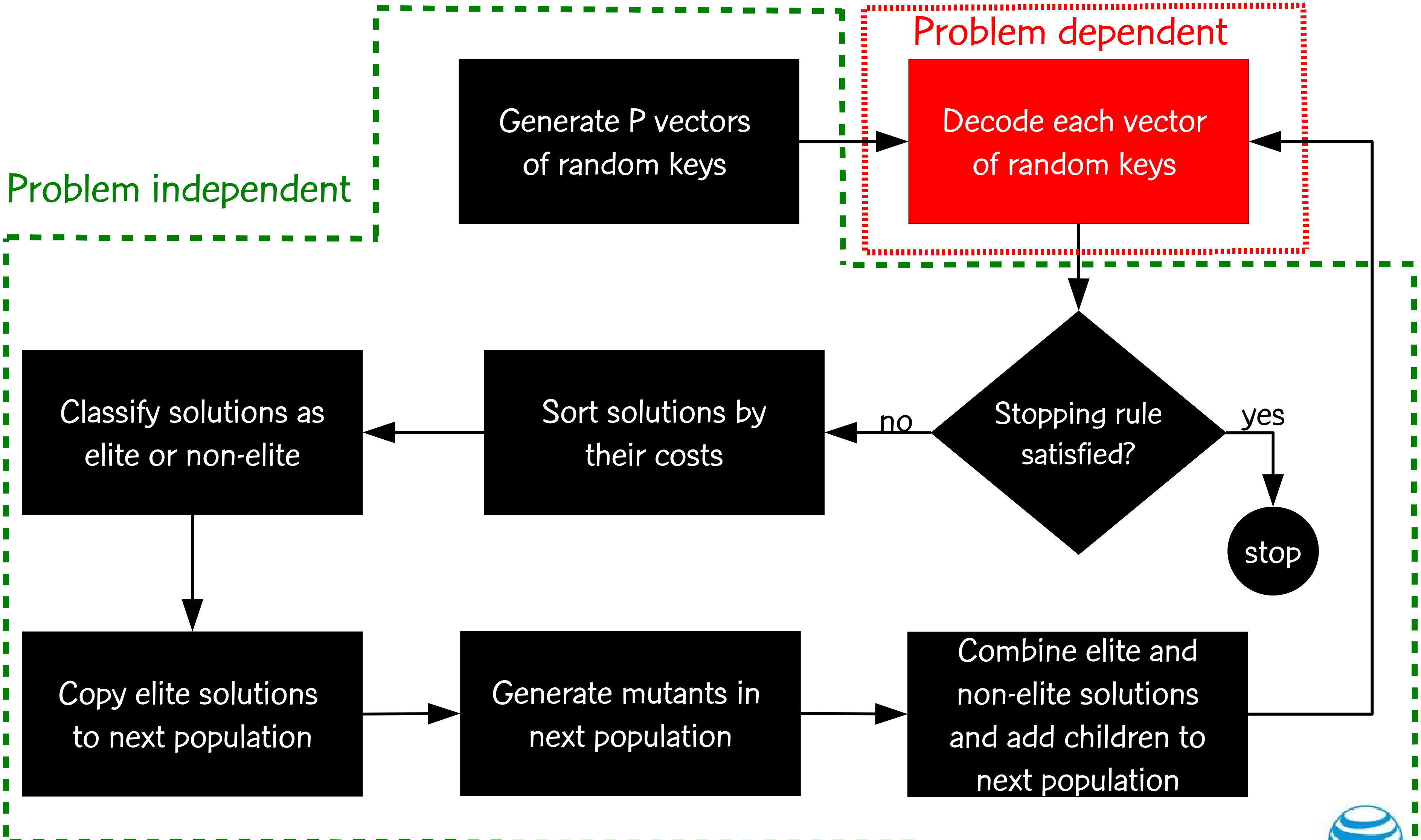


Framework for random-key genetic algorithms



at&t
Your world. Delivered.

Framework for random-key genetic algorithms



GA for the MMS problem

GA for the MMS problem

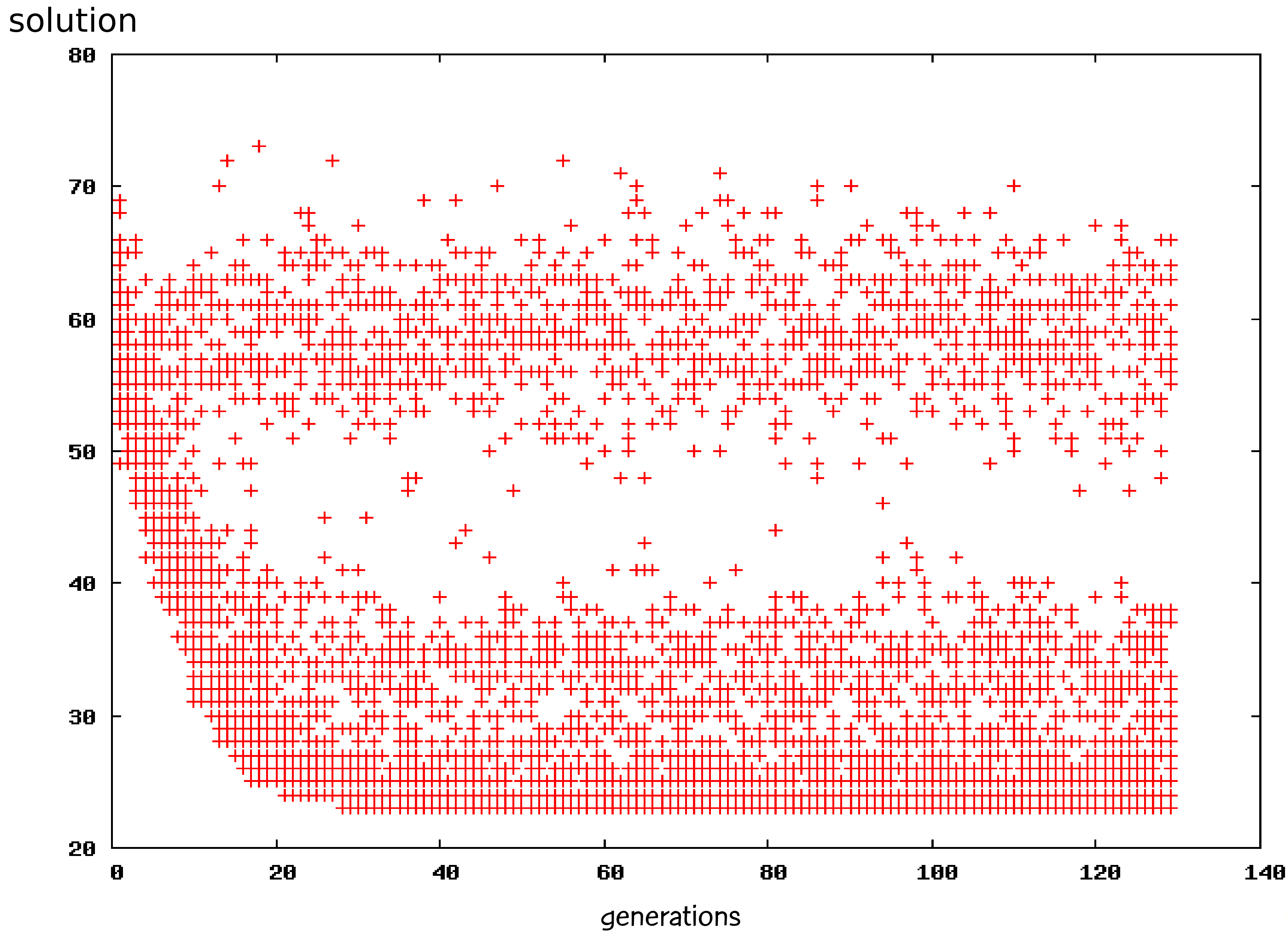
- Chromosome:
  - A vector X of N random 0-1values (random keys), where N is the number of potential monitoring nodes. The i-th random key corresponds to the i-th monitoring node.
- Decoder:
  - For i = 1,N: if X(i) = 1, add i-th monitoring node to solution
  - If solution is feasible, i.e. all customer nodes are covered: STOP
  - Else, apply greedy algorithm to cover uncovered branch nodes.

at&t
Your world. Delivered.

GA for the MMS problem

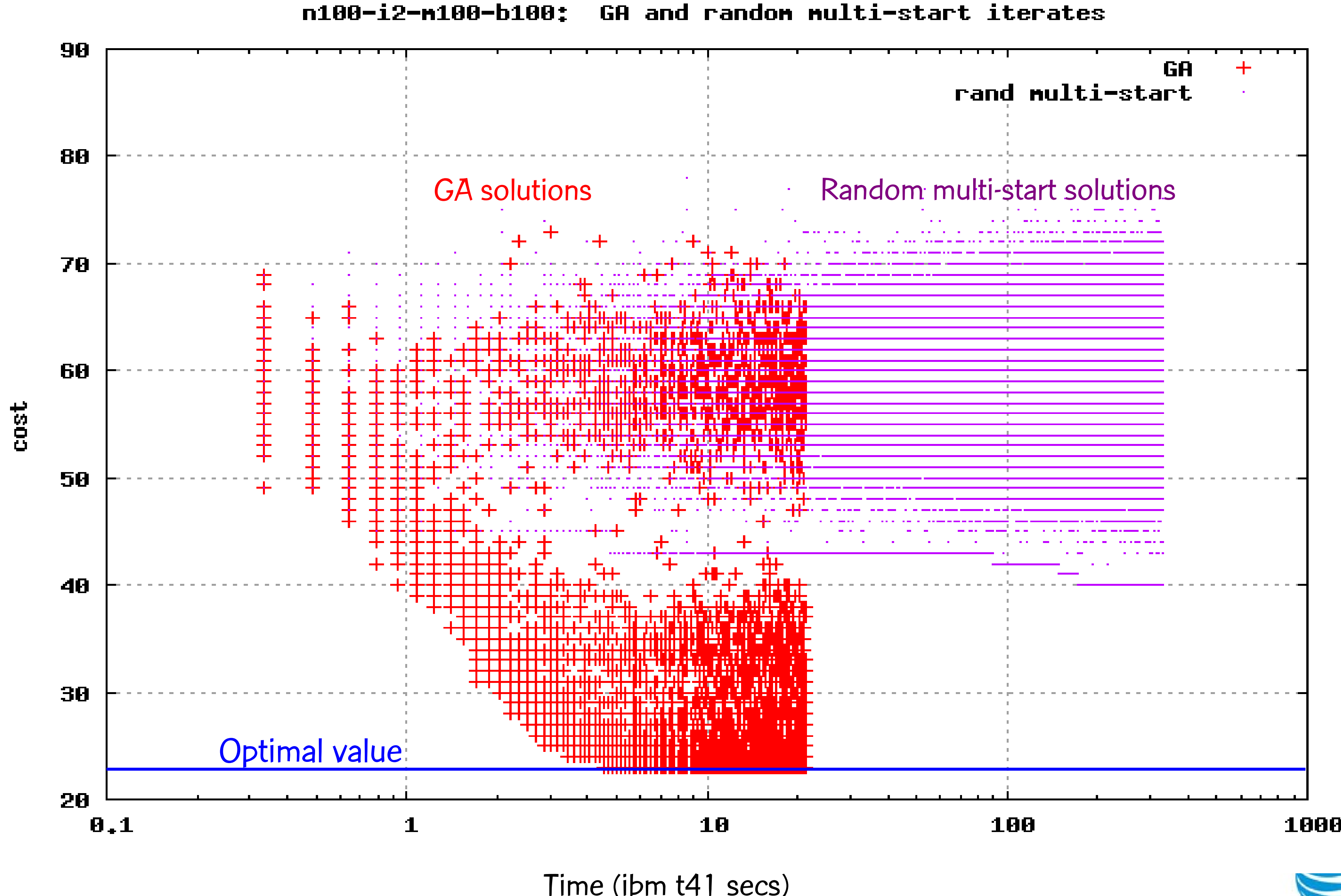
- Size of population: N (number of monitoring nodes)
- Size of elite set: 15% of N
- Size of mutant set: 10% of N
- Biased coin probability: 70%
- Stop after N generations without improvement of best found solution

n100-i2-m100-b100 (opt = 23)

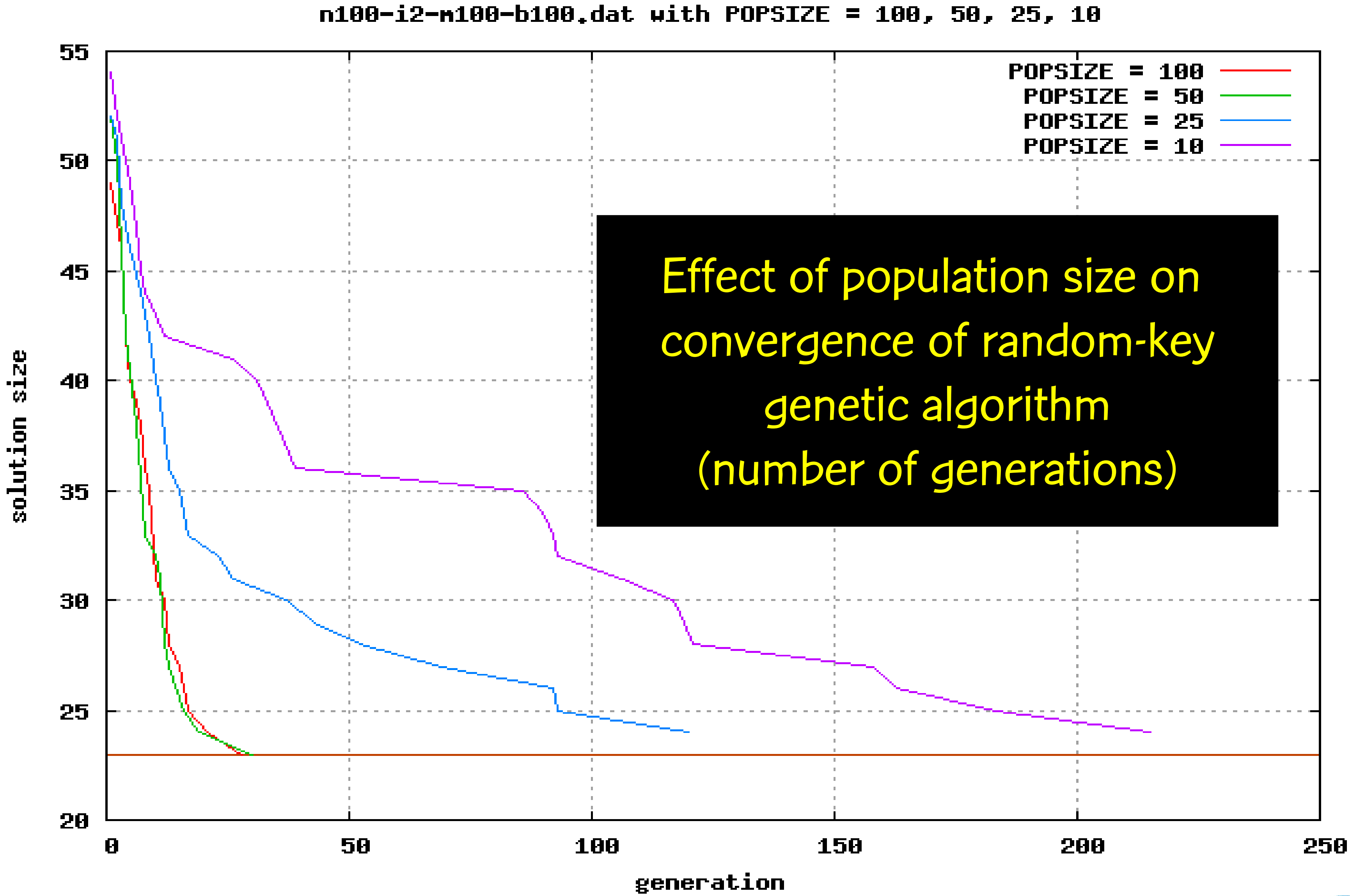


n100-i2-m100-b100 (opt = 23)

solution

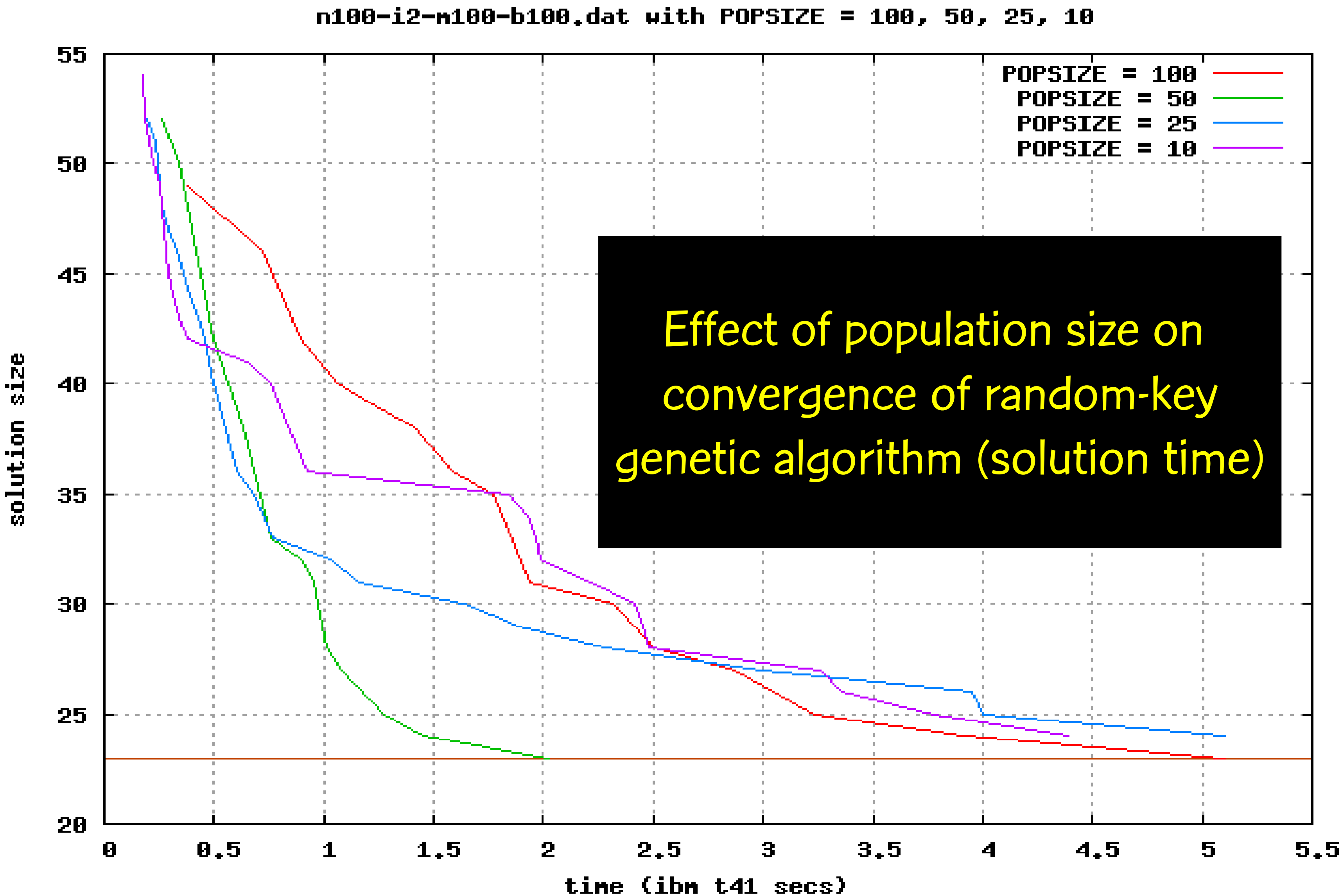


n100-i2-m100-b100 (opt = 23)

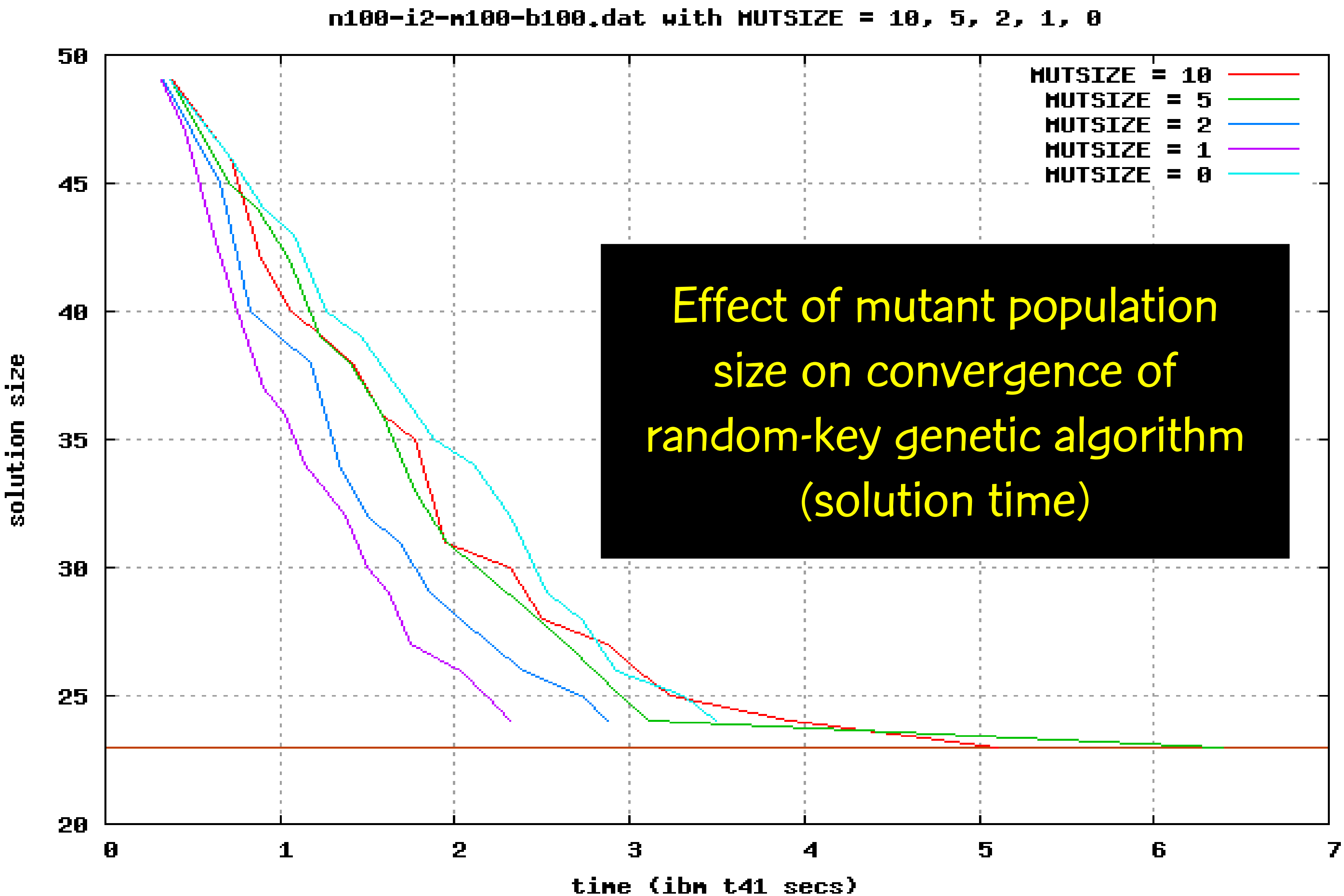


at&t
Your world. Delivered.

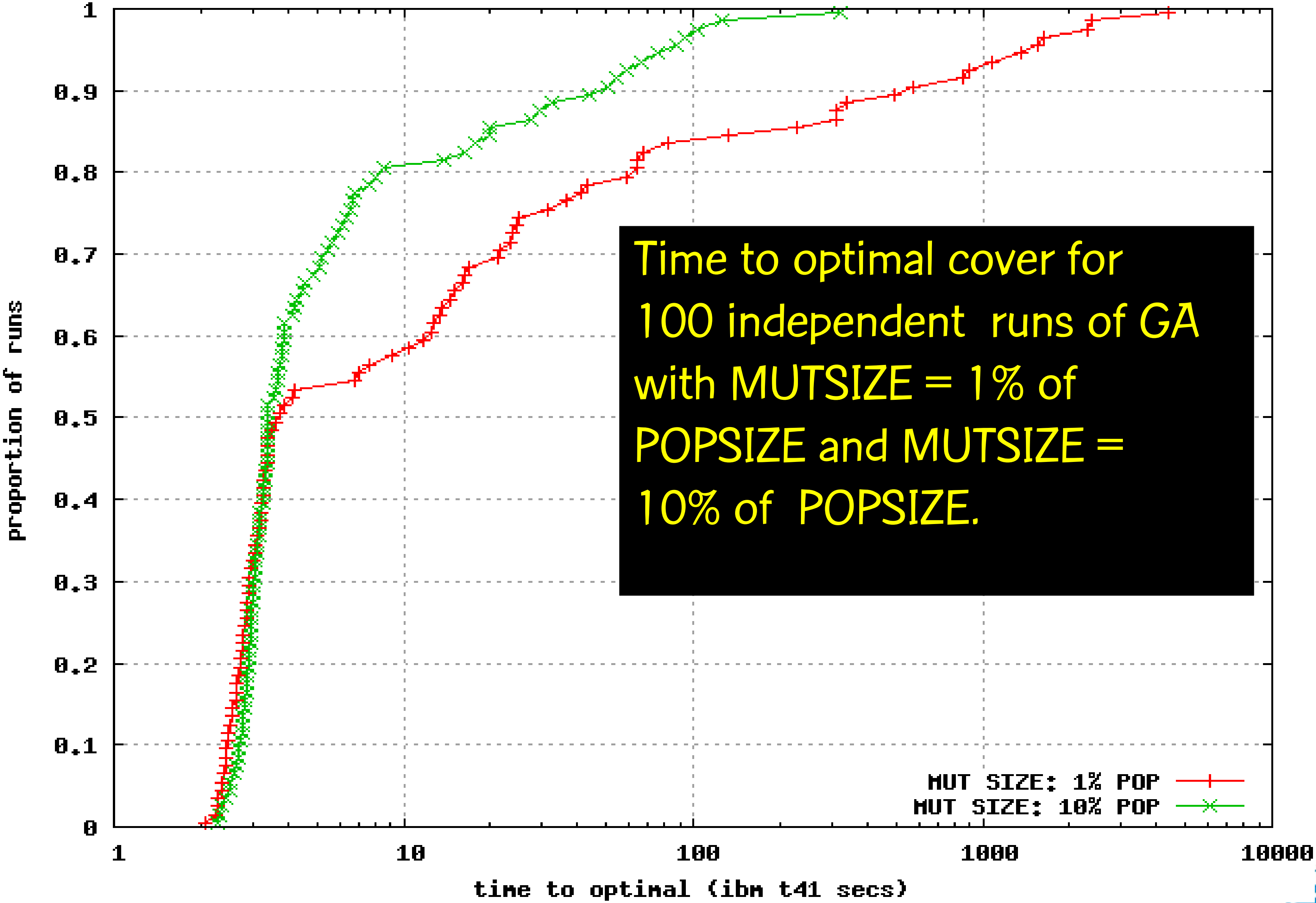
n100-i2-m100-b100 (opt = 23)



n100-i2-m100-b100 (opt = 23)



n100-i2-m100-b100 (opt = 23)



Time to optimal cover for 100 independent runs of GA with MUTSIZE = 1% of POPSIZE and MUTSIZE = 10% of POPSIZE.

Experimental results

- 560 instances, with 25, 50, 100, 190, 220, 250, 300, and 558 nodes.
- 324 of these 560 were solved optimally with CPLEX. Running GA a single time, we found optimal solutions in 318 of these instances.
- Of the 236 that CPLEX could not solve, GA matched a lower bound in 166.
- In all, the GA found optimal solutions for 484 of the 560 instances (86.4%)

Experimental results

- The paper describes the double hitting set heuristic (HH). This heuristic makes use of the OSPF paths and is very fast and effective.
- In 482 of the 560 instances (86.1%) the GA and HH found solutions with the same cost.
- In 68 instances (12.1%) GA found a better solution than HH.
- In 10 instances (2%) HH found a better solution than GA.
- In only 12 instances (2.1%) was the solution found by GA not minimal.

Concluding remarks

at&t
Your world. Delivered.

Concluding remarks

- We constructed a number of network test instances to capture the topology and routing of large internetworks;
- We demonstrated algorithms that provide a feasible combination of accuracy and execution times;
- We showed that solutions derived from our methods provide a useful saving in the number of measurement nodes compared with the naive approach of using each branch point as a measurement node: Networks having a large number of branch nodes need only 20-30% of branch points to be measurement nodes.

The End

These slides and all of my papers cited in this talk can be downloaded from my homepage: http://mauricioresende.com