A memetic algorithm for routing optimization in networks using exponential flow splitting

Talk given at the Ninth INFORMS Telecommunications Conference, University of Maryland, College Park, MD
March 28, 2008

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

Joint work with Luciana Buriol,
Marcus Ritt, and Roger Reis ~ UFRGS, Brazil.

Summary of talk

- Genetic algorithm with random keys
- OSPF routing
- Memetic algorithm for OSPF routing
- DEFT routing
- Implementation issues
- Numerical results for OSPF and DEFT
- Concluding remarks

Genetic algorithms with 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)
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

at&t
Your world. Delivered.

Informática
UFRGS

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)

at&t
Your world. Delivered.

Informática
UFRGS

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)

at&t
Your world. Delivered.

Informática
UFRGS

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.

Informática
UFRGS

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)

at&t
Your world. Delivered.

Informática
UFRGS

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)

at&t
Your world. Delivered.

Informática
UFRGS

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)

at&t
Your world. Delivered.

Informática
UFRGS

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)

at&t
Your world. Delivered.

Informática
UFRGS

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)

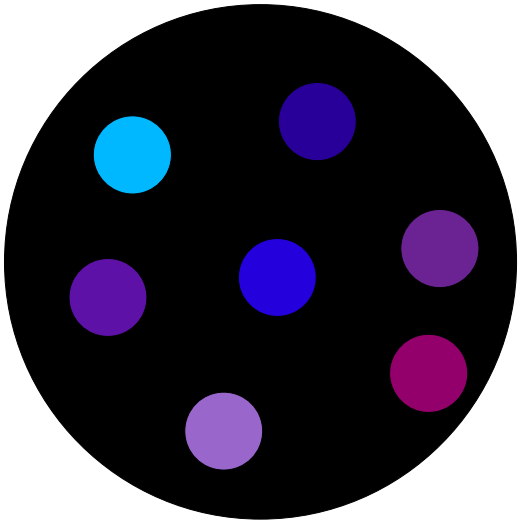
Every random-key array corresponds to a feasible solution: Mating always produces feasible offspring.

at&t
Your world. Delivered.

Informática
UFRGS

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].

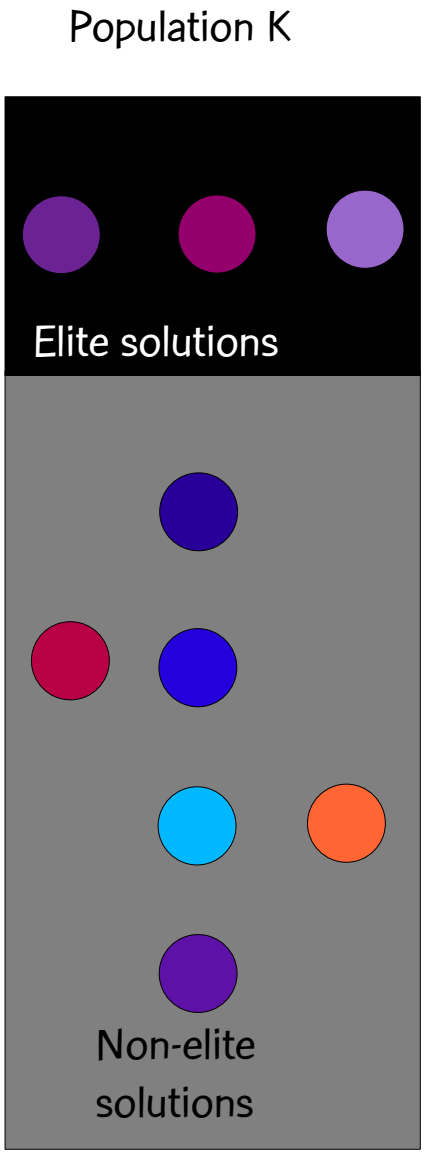


at&t
Your world. Delivered.

Informática
UFRGS

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.

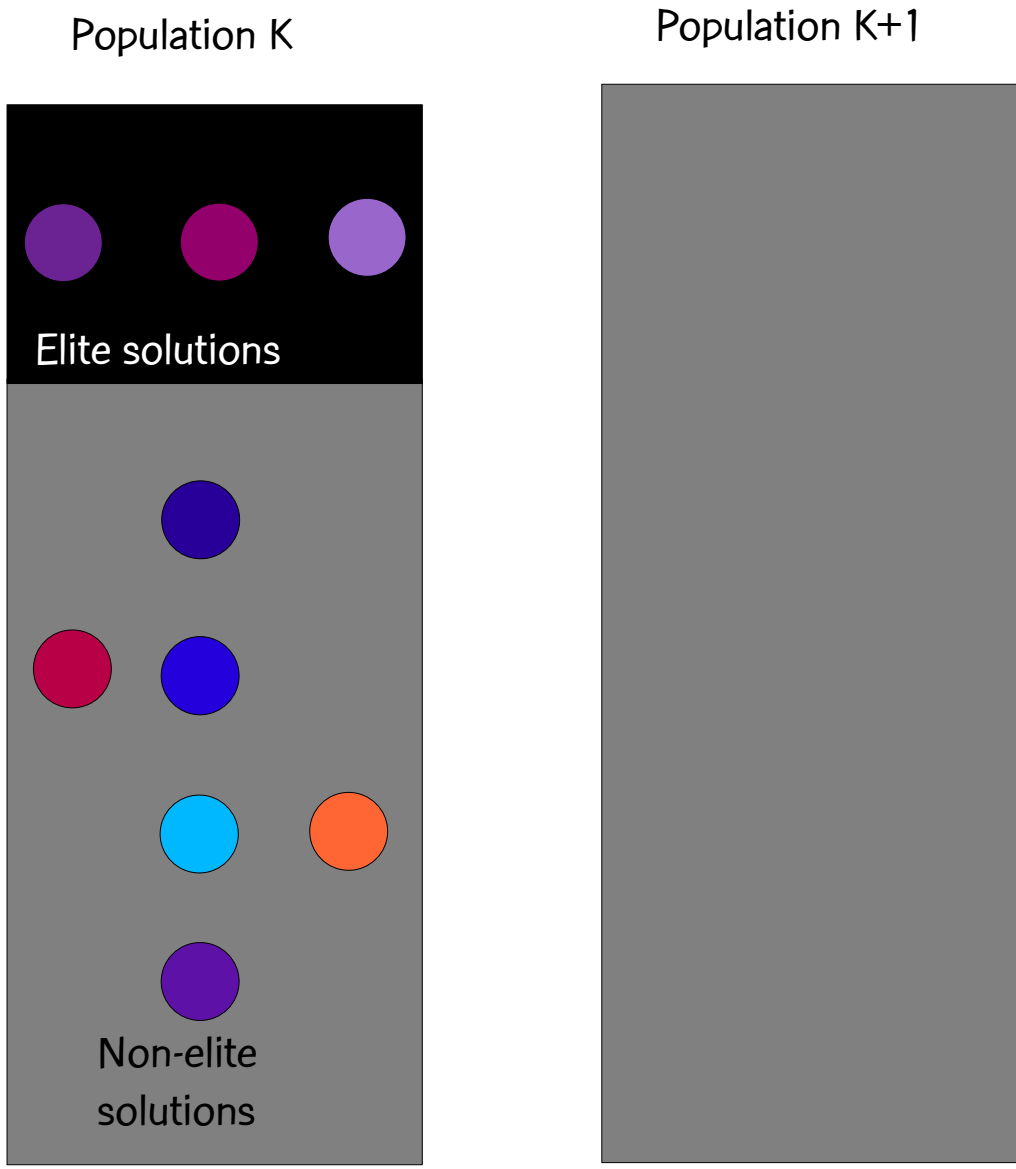


at&t
Your world. Delivered.

Informática UFRGS

GAs and random keys

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

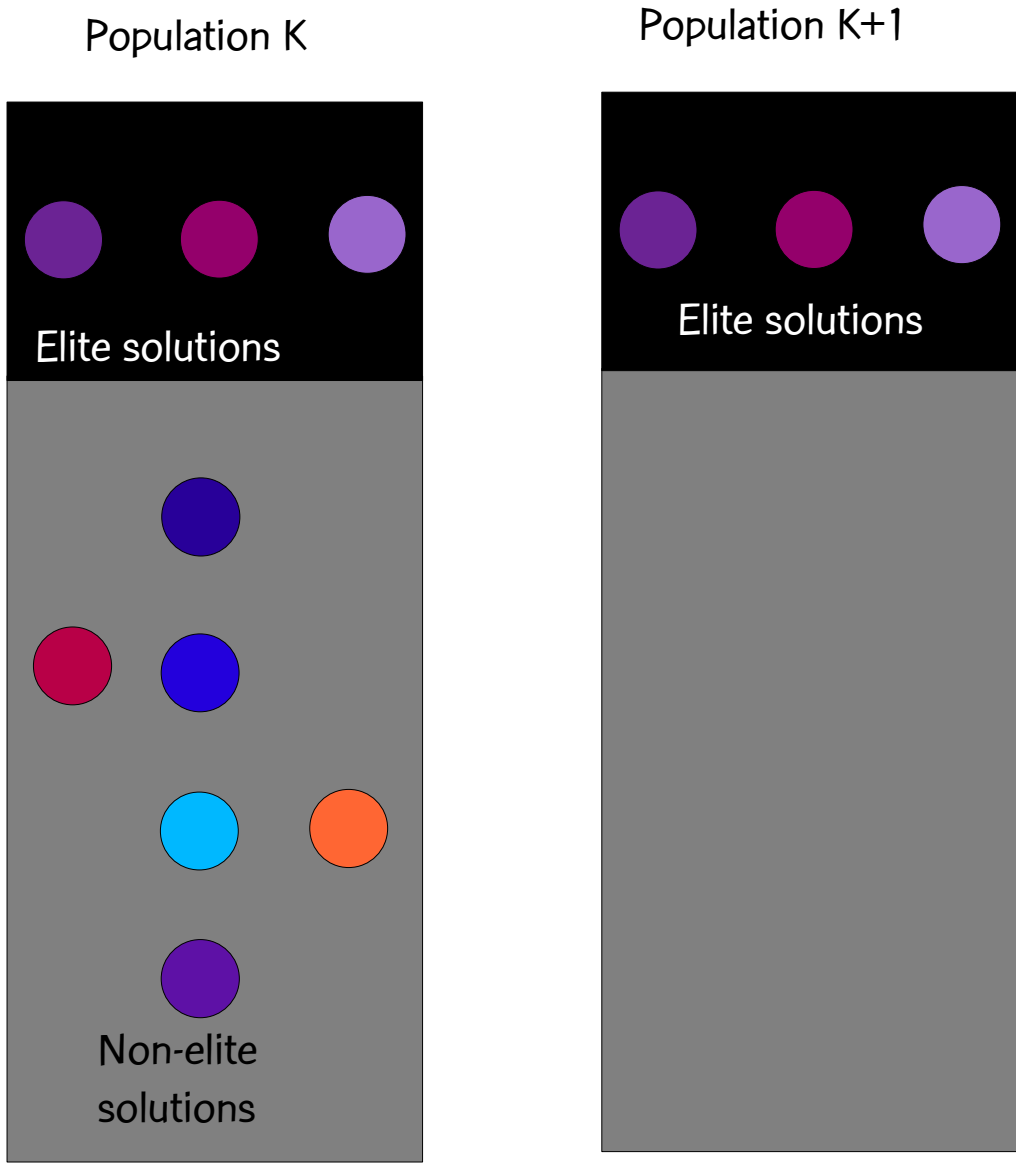


at&t
Your world. Delivered.

Informática UFRGS

GAs and random keys

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

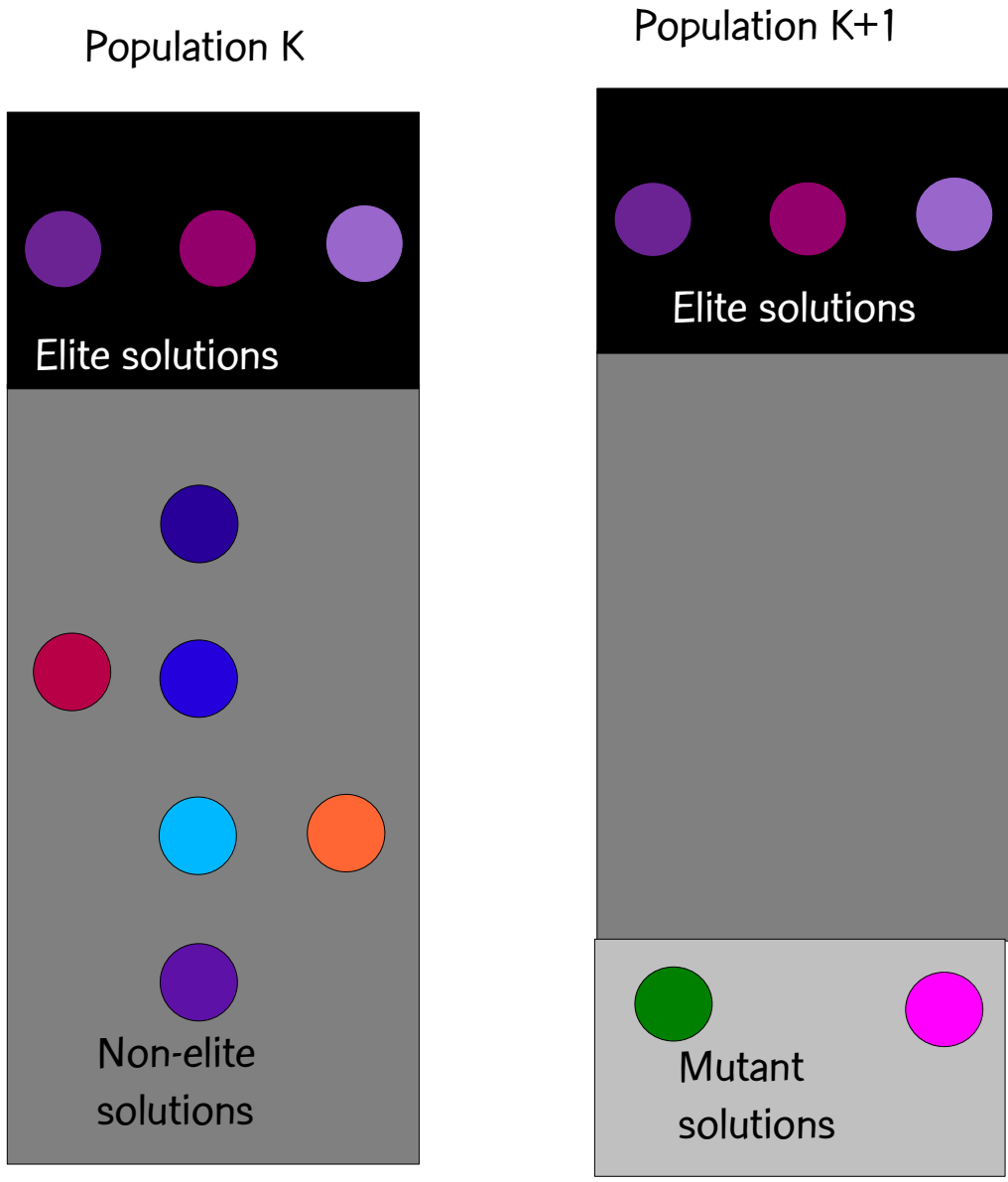


at&t
Your world. Delivered.

Informática UFRGS

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

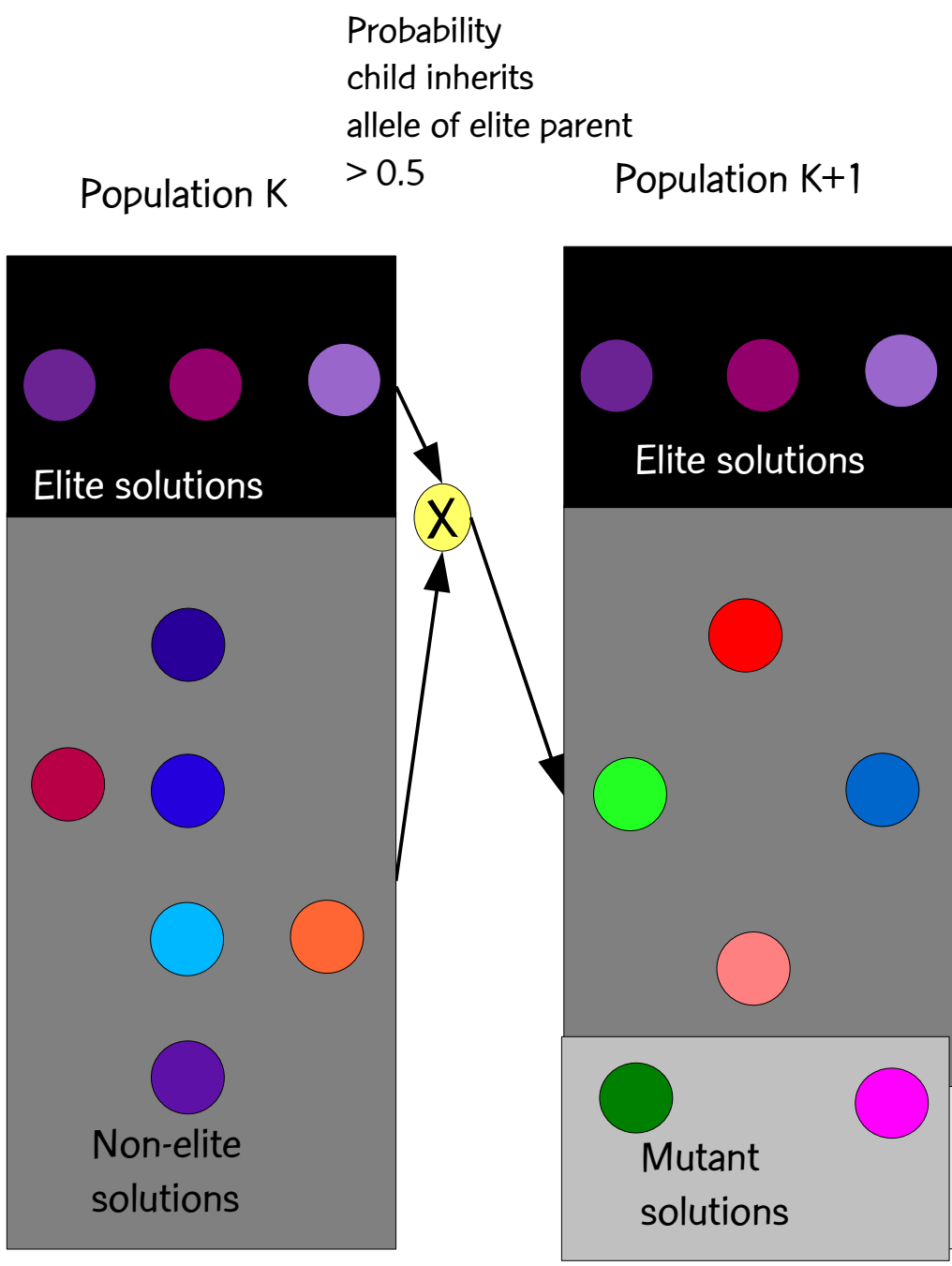


at&t
Your world. Delivered.

Informática UFRGS

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.

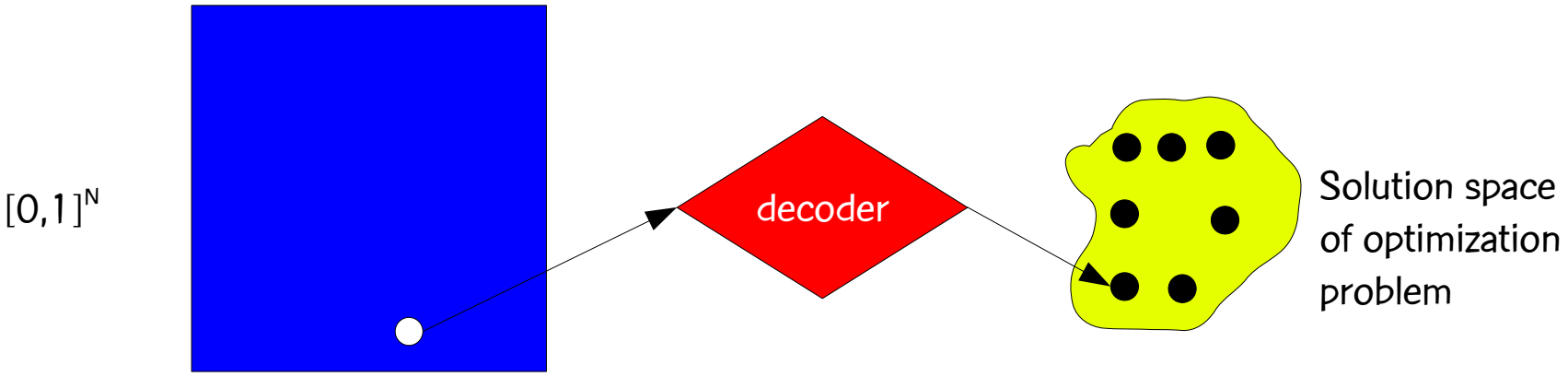


at&t
Your world. Delivered.

Informática
UFRGS

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.

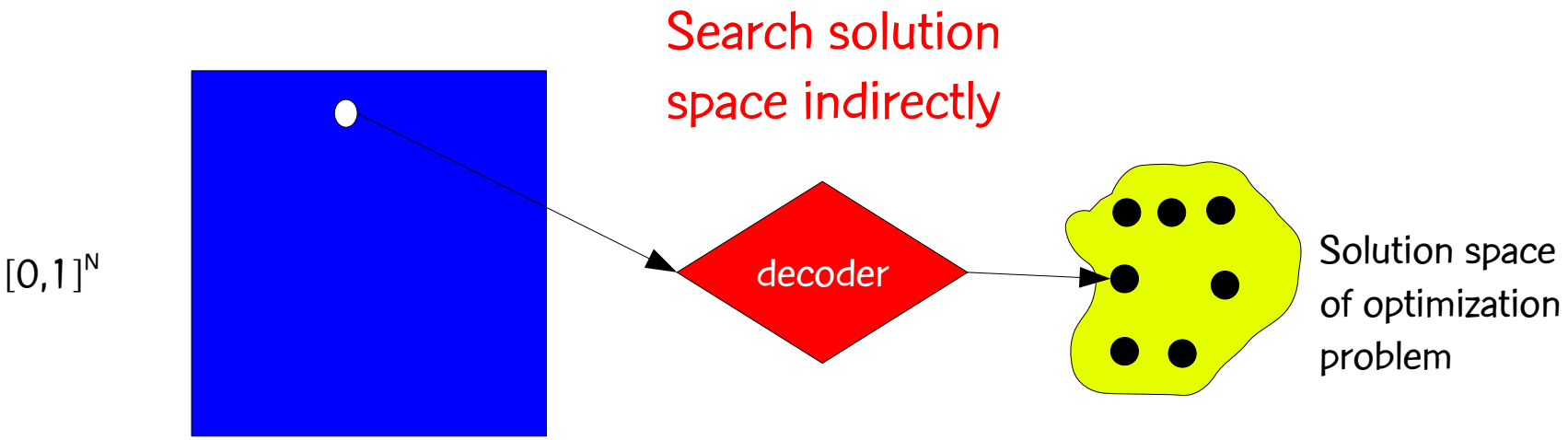


at&t
Your world. Delivered.

Informática
UFRGS

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.

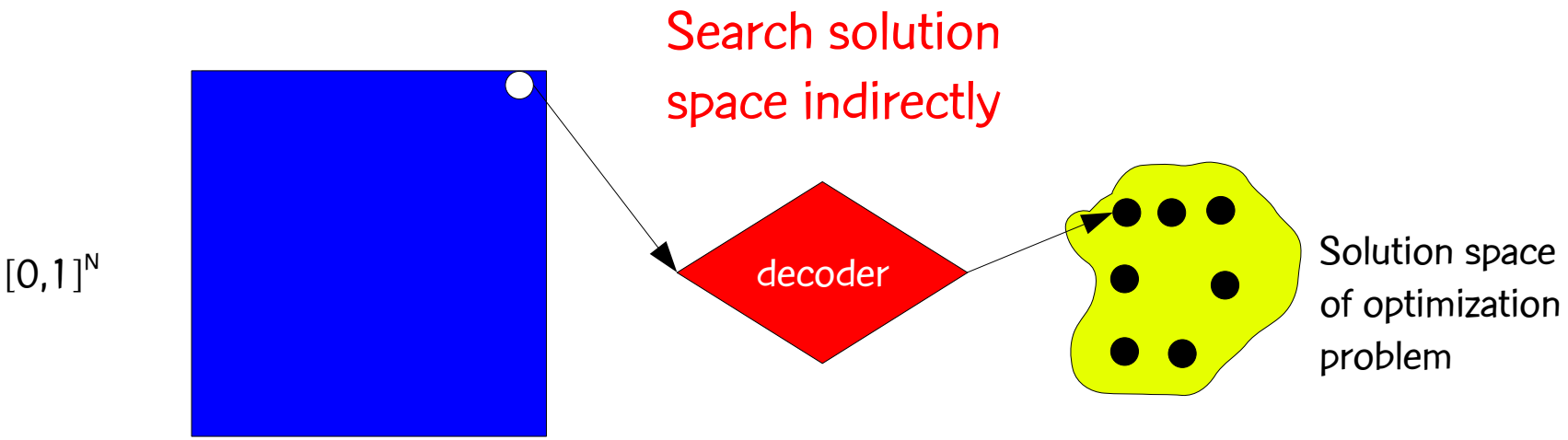


at&t
Your world. Delivered.

Informática
UFRGS

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.

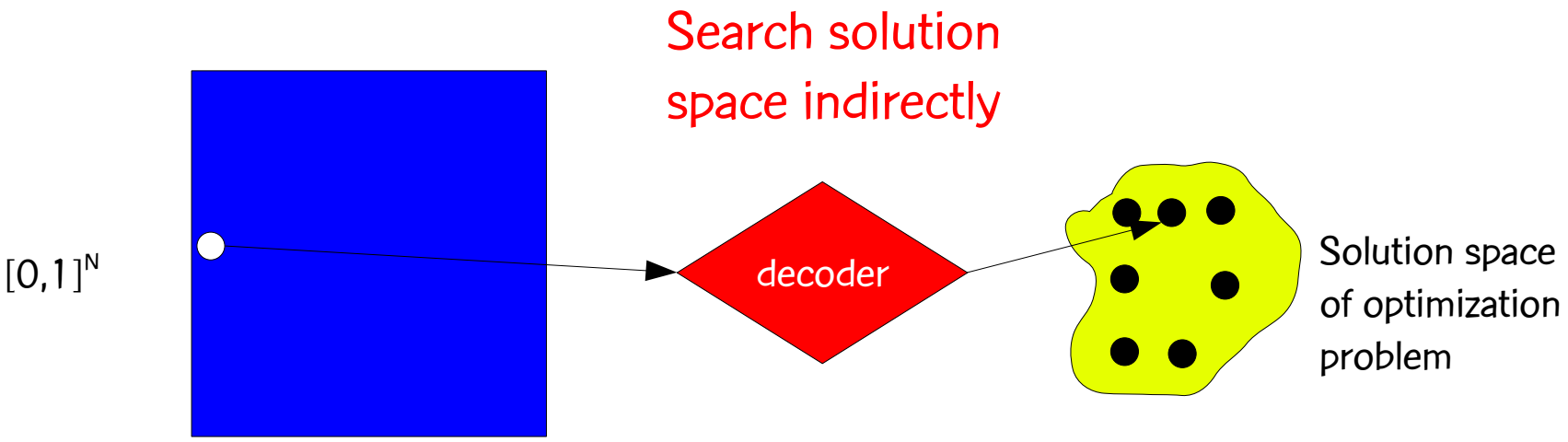


at&t
Your world. Delivered.

Informática
UFRGS

Decoders

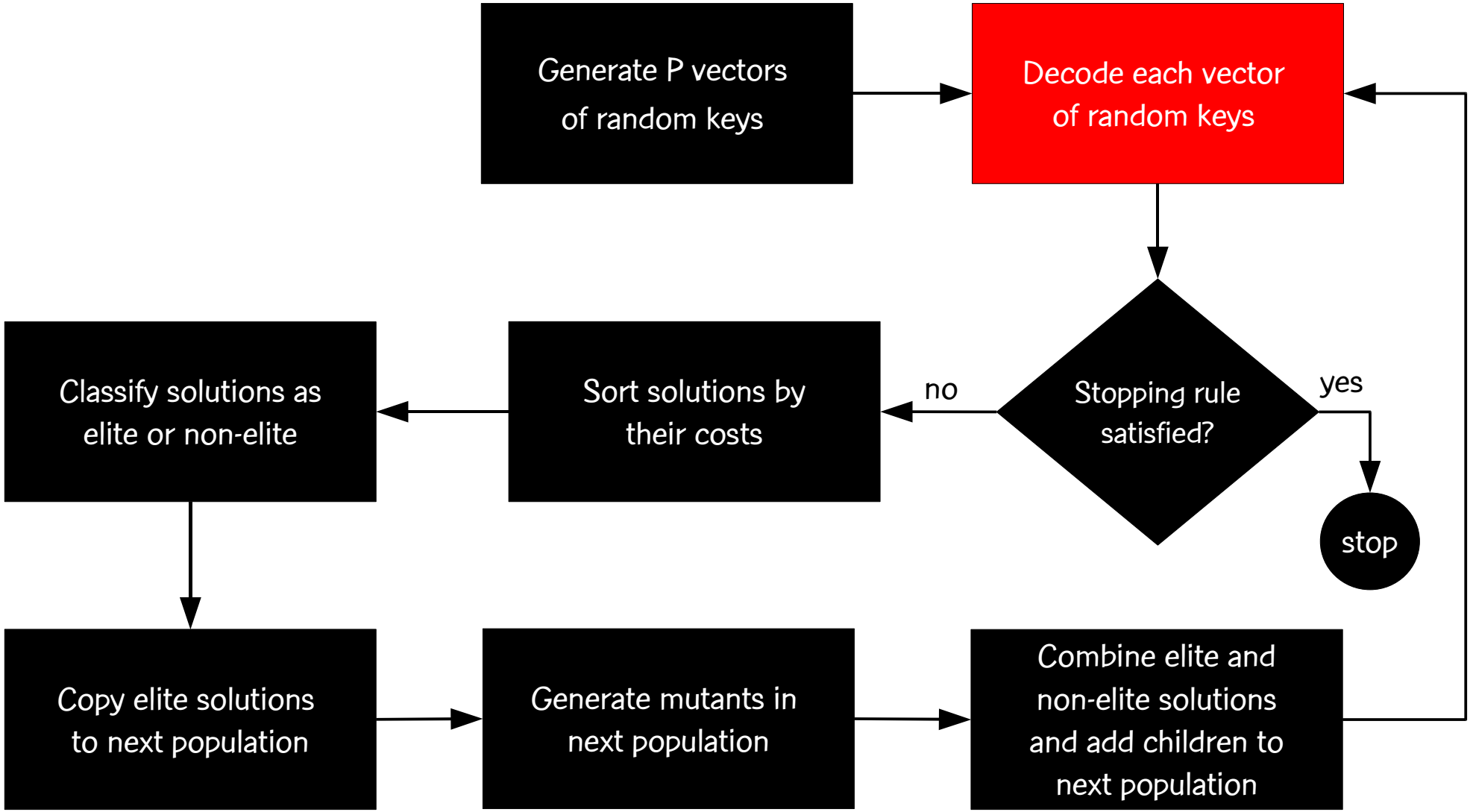
- A decoder is a deterministic algorithm that takes as input a random-key vector and returns a feasible solution of the optimization problem and its cost.
- Bean (1994) proposed decoders based on sorting the random-key vector to produce a sequence.
- A random-key GA searches the solution space indirectly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



at&t
Your world. Delivered.

Informática
UFRGS

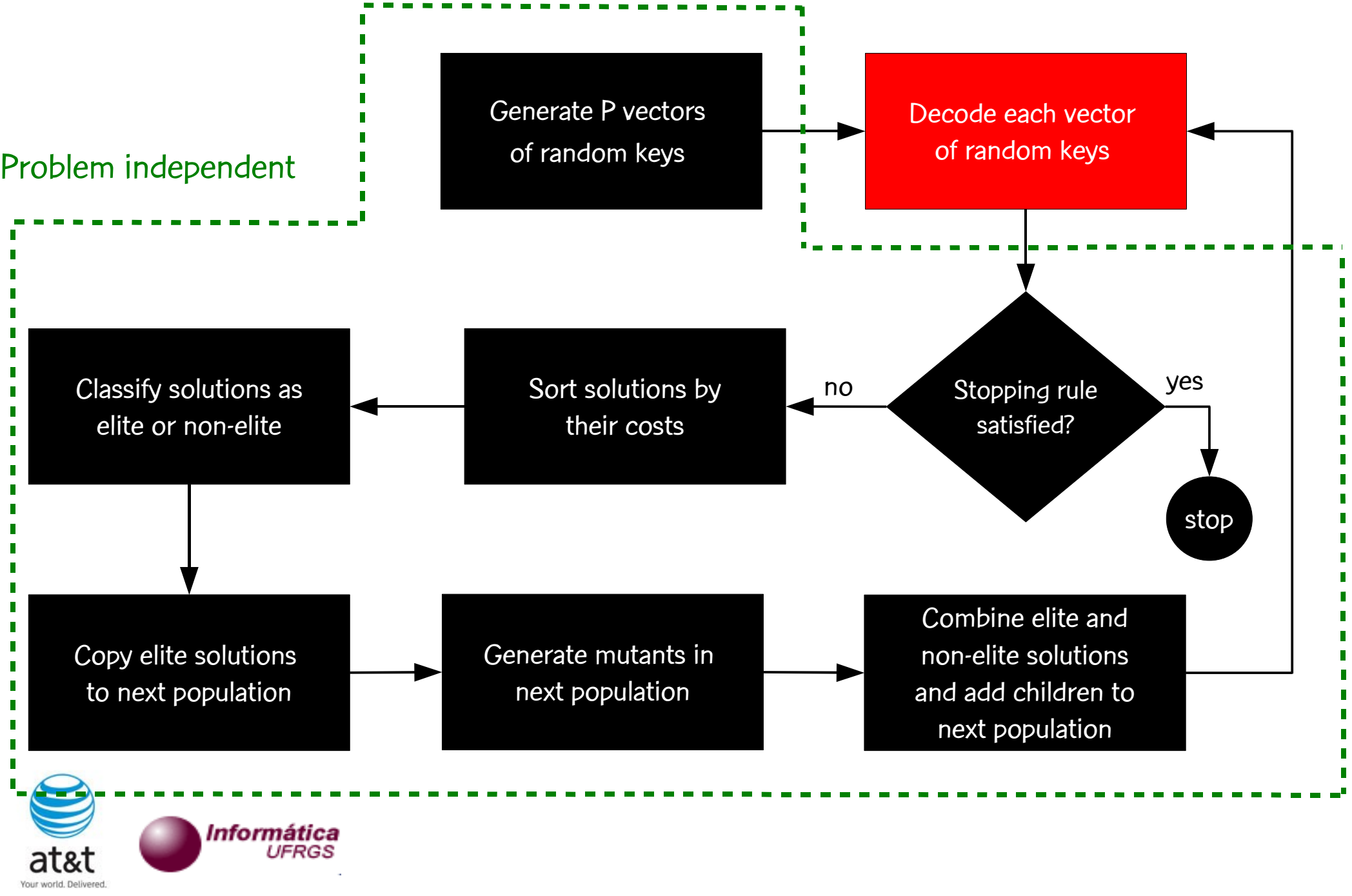
Framework for random-key genetic algorithms



at&t
Your world. Delivered.

Informática UFRGS

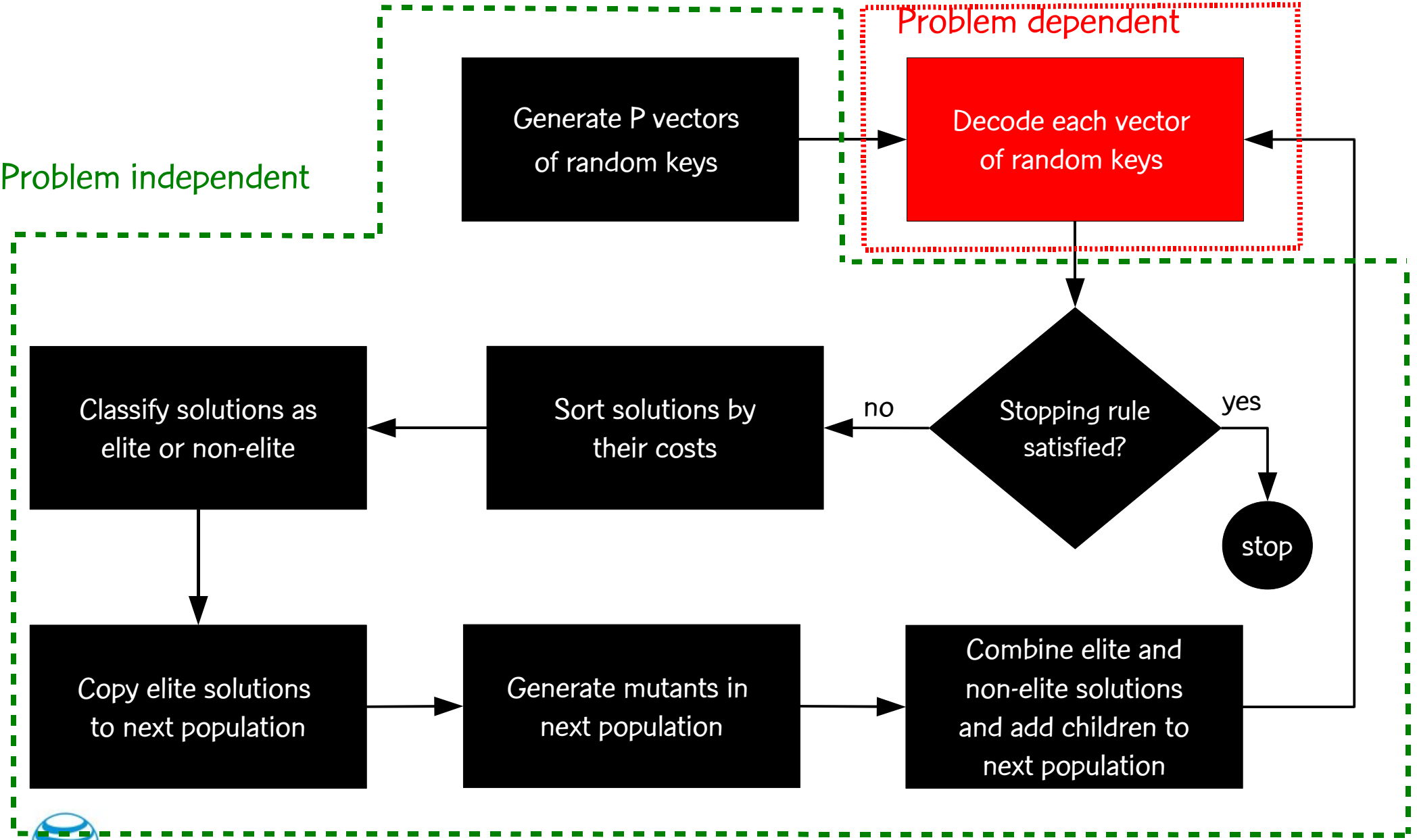
Framework for random-key genetic algorithms



at&t
Your world. Delivered.

Informática UFRGS

Framework for random-key genetic algorithms



at&t
Your world. Delivered.

Informática UFRGS

OSPF routing in IP networks

Routing in IP networks

- Protocol: In OSPF, traffic is routed on shortest weight paths from origination router to destination router.
- Splitting: If more than one link out of a router is on a shortest weight path, traffic is evenly distributed on those links.
- Weight setting problem: Determine OSPF weights such that if traffic is routed according to OSPF protocol, network congestion is minimized.

Minimization of congestion

- Consider the directed capacitated network $G = (N, A, c)$, where N are routers, A are links, and c_a is the capacity of link $a \in A$.
- We use the measure of Fortz & Thorup (2000) to compute congestion:

$$\Phi = \Phi_1(l_1) + \Phi_2(l_2) + \ldots + \Phi_{|A|}(l_{|A|})$$

where l_a is the load on link $a \in A$,

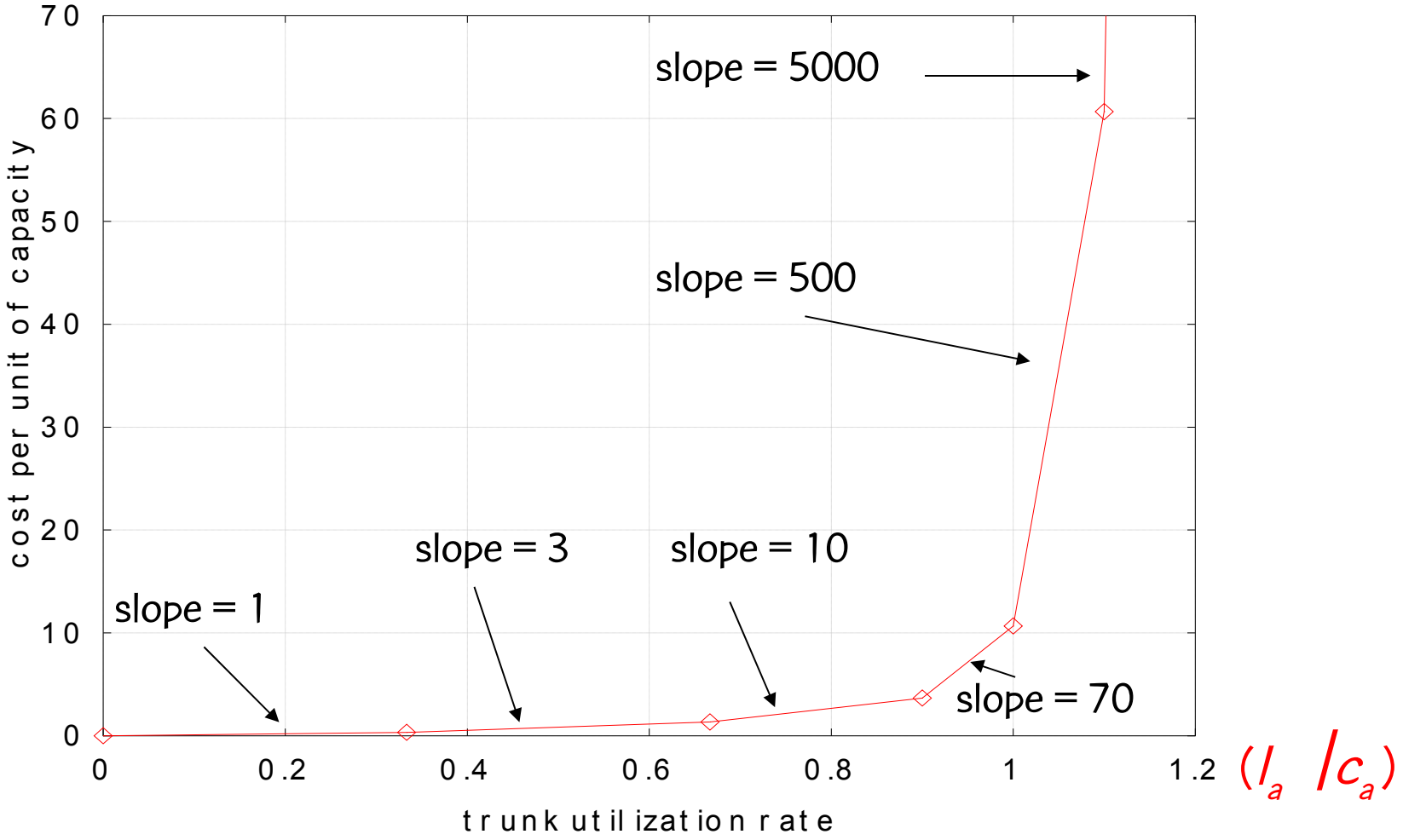
$\Phi_a(l_a)$ is piecewise linear and convex,

$\Phi_a(0) = 0$, for all $a \in A$.

at&t
Your world. Delivered.

Informática
UFRGS

Piecewise linear and convex $\Phi_a(I_a)$ link congestion measure



at&t
Your world. Delivered.

Informática UFRGS

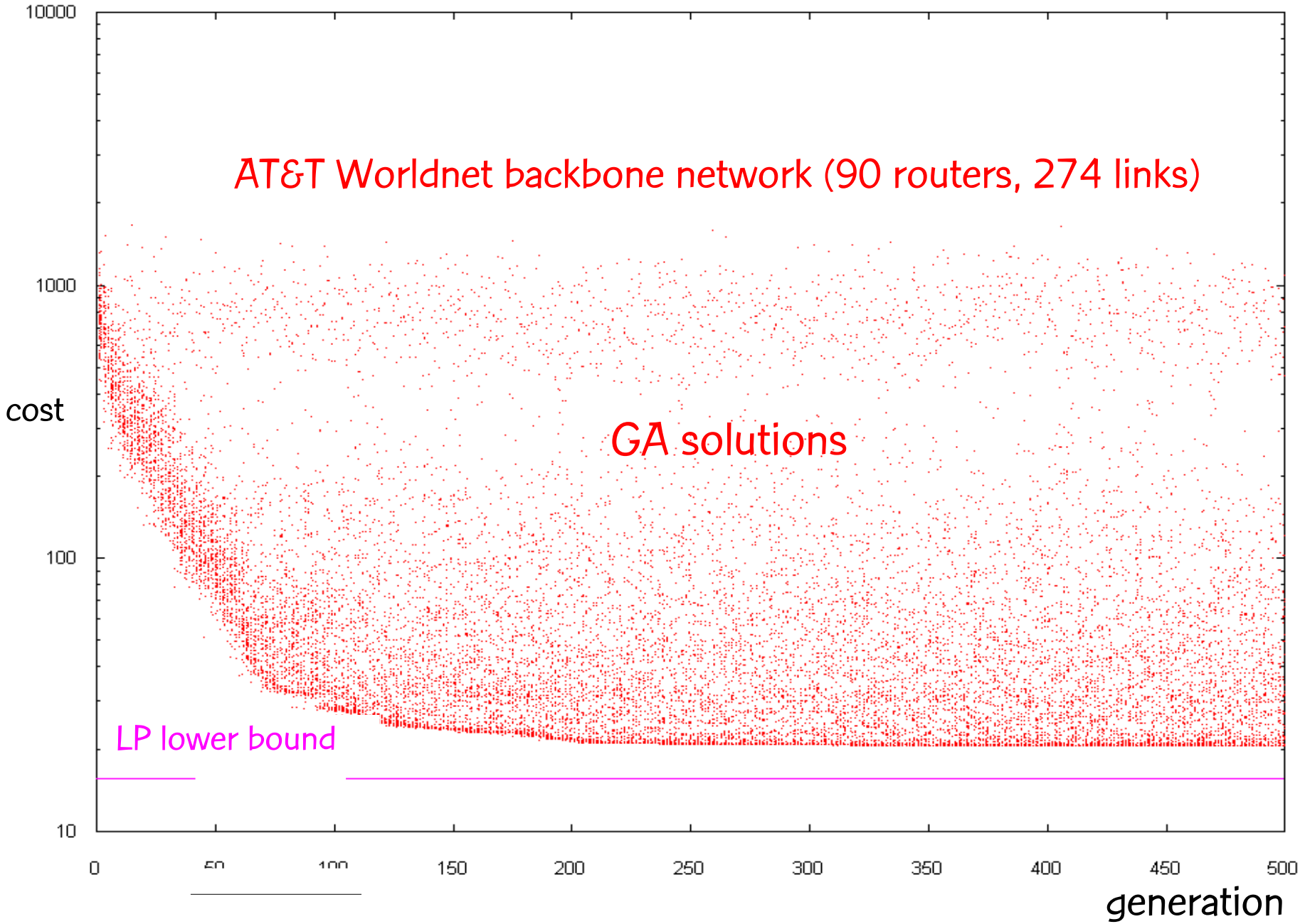
Genetic algorithm for OSPF routing in IP networks

Ericsson, R., & Pardalos (J. Comb. Opt., 2002)

- Chromosome:
  - A vector X of N random keys, where N is the number of links. The i-th random key corresponds to the i-th link weight.
- Decoder:
  - For i = 1,N: set w(i) = ceil (X(i) $\times$ w_{max})
  - Compute shortest paths and route traffic according to OSPF.
  - Compute load on each link, compute link congestion, add up all link congestions to compute network congestion.

at&t
Your world. Delivered.

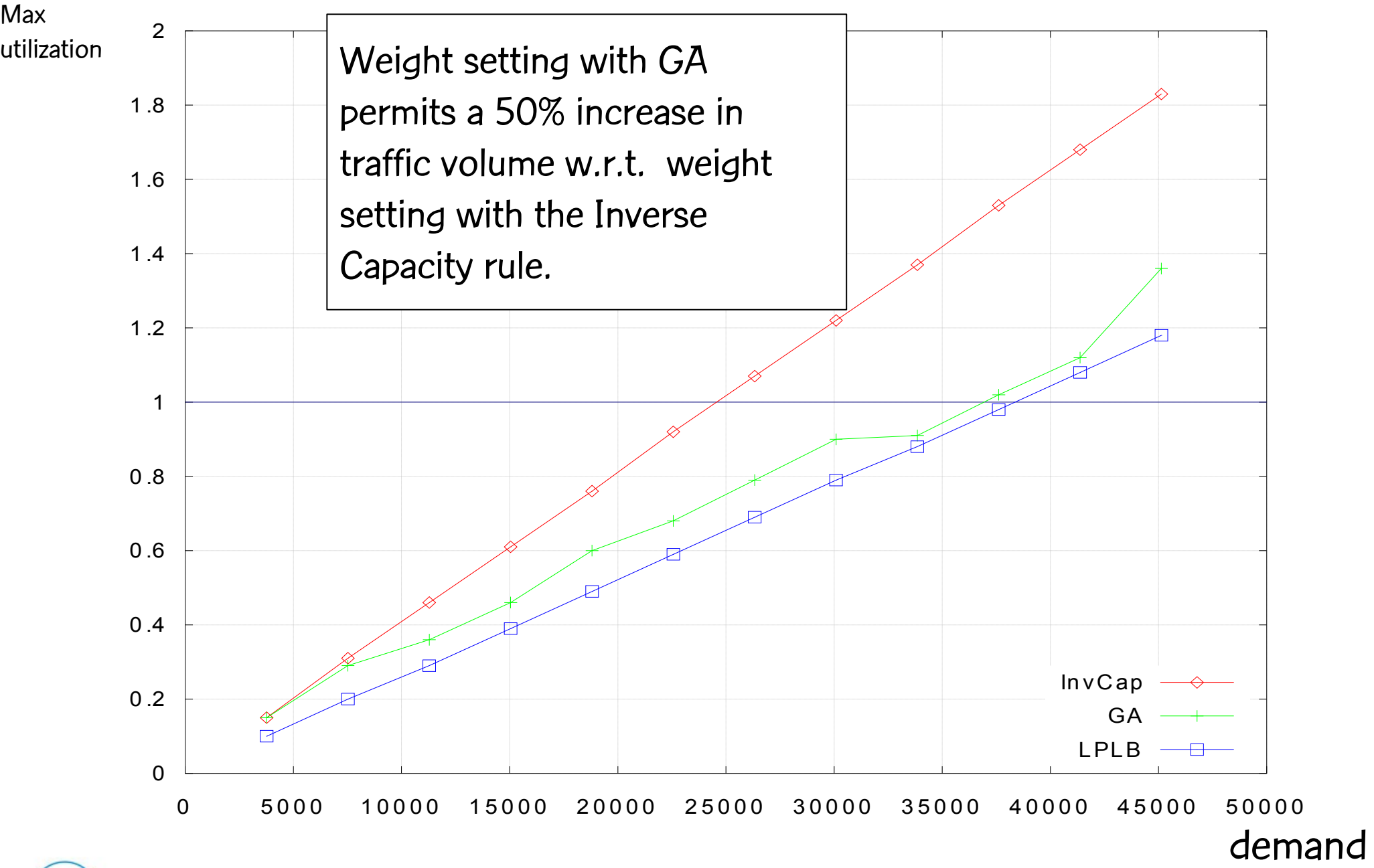
Informática
UFRGS



at&t
Your world. Delivered.

Informática
UFRGS

AT&T Worldnet backbone network (90 routers, 274 links)



Weight setting with GA permits a 50% increase in traffic volume w.r.t. weight setting with the Inverse Capacity rule.

at&t
Your world. Delivered.

Informática UFRGS

Memetic algorithms for OSPF routing in IP networks

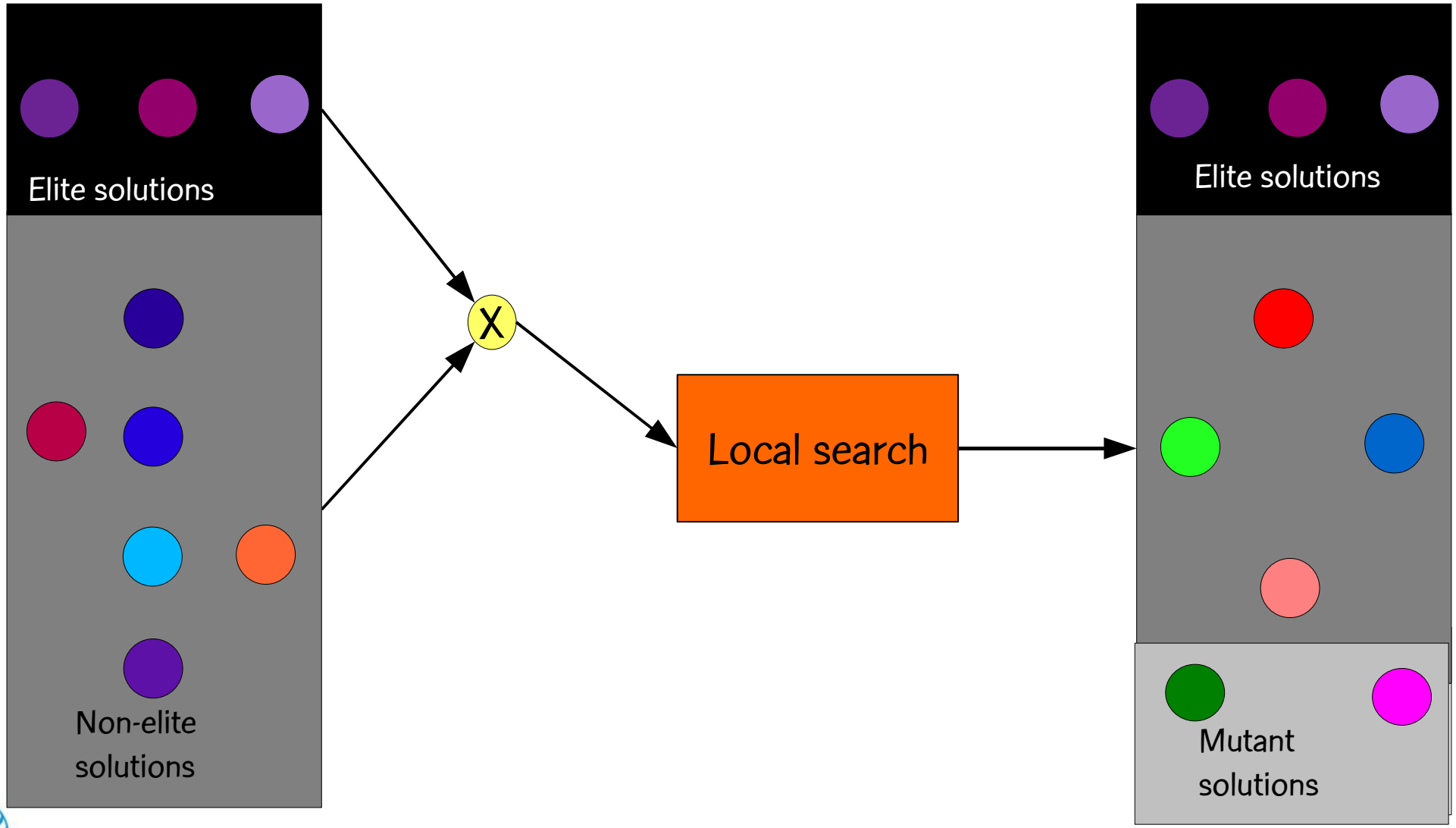
Buriol, R., Ribeiro, and Thorup (Networks, 2005)

- Chromosome:
  - A vector X of N random keys, where N is the number of links. The i-th random key corresponds to the i-th link weight.
- Decoder:
  - For i = 1,N: set $w(i) = ceil\ (\ X(i) \times w_{max}\ )$
  - Compute shortest paths and route traffic according to OSPF.
  - Compute load on each link, compute link congestion, add up all link congestions to compute network congestion.
  - Apply fast local search to improve weights.

at&t
Your world. Delivered.

Informática
UFRGS

Memetic algorithm: Optimized crossover = crossover + local search



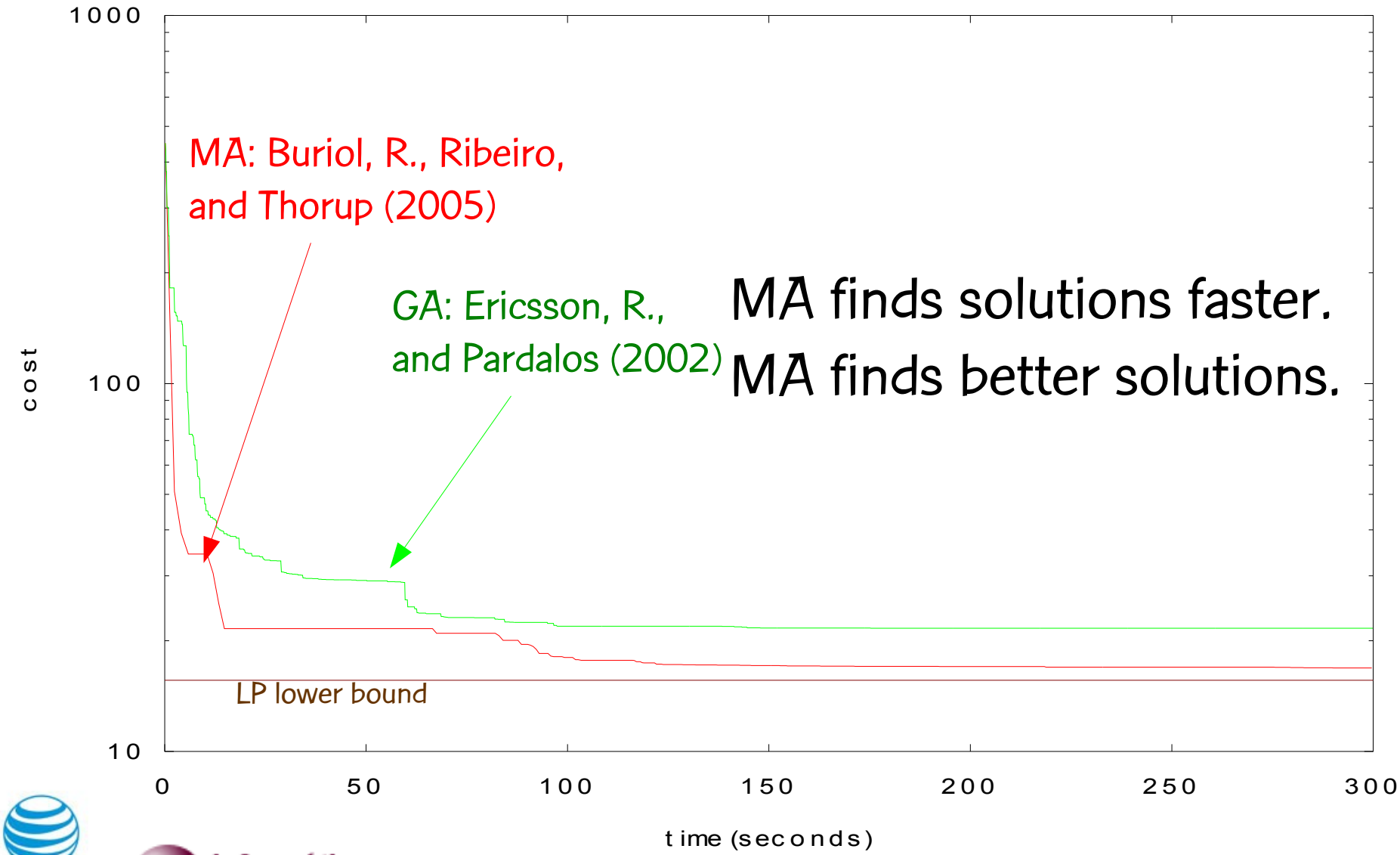
at&t
Your world. Delivered.

Informática UFRGS

Fast local search

- Let A^* be the set of five arcs $a \in A$ having largest Φ_a values.
- Scan arcs $a \in A^*$ from largest to smallest Φ_a:
  - Increase arc weight, one unit at a time, in the range $[w_a , w_a + \lceil (w_{max} - w_a)/4 \rceil]$
  - If total cost Φ is reduced, restart local search.

Effect of decoder with fast local search



at&t
Your world. Delivered.

Informática UFRGS

DEFT routing in IP networks

DEFT routing

- Proposed by Dahai Xu, Mung Chiang, and Jennifer Rexford, DEFT: Distributed Exponentially-weighted Flow spliTting, INFOCOM 2007

- Flow is routed on all links that lead to the destination. An *exponential penalty* is used to assign less flow to links that are on longer paths.

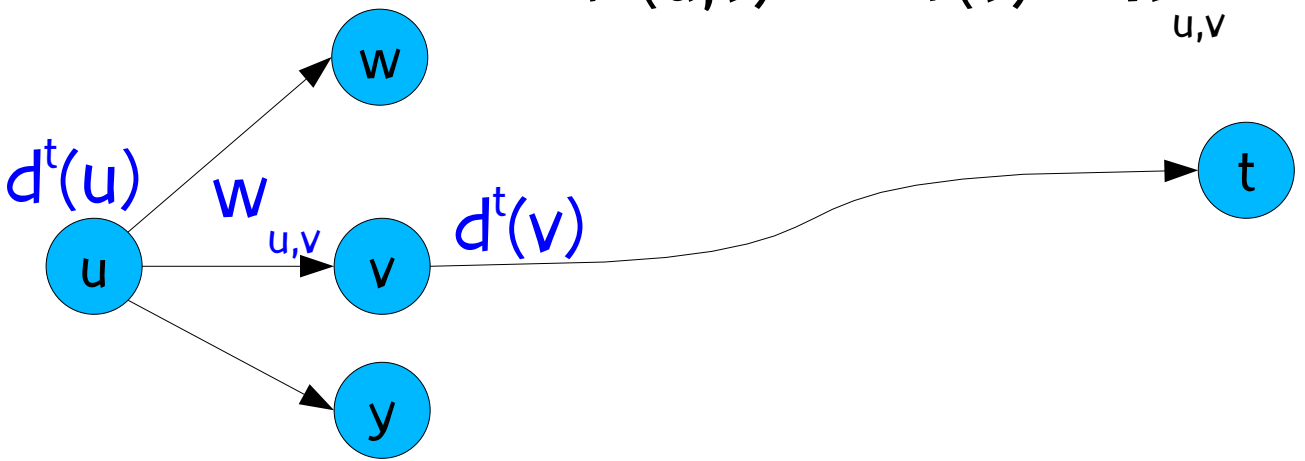
at&t
Your world. Delivered.

Informática
UFRGS

DEFT routing

- Consider each forward link (u,v) outgoing a given node u.
- Denote by $w_{u,v}$ the real-valued weight of link (u,v) and $d^t(u)$ as the distance of node u from target t.
- The gap $h^t(u,v)$ between u and v is calculated as:

$$h^t(u,v) = d^t(v) + w_{u,v} - d^t(u)$$



at&t
Your world. Delivered.

Informática UFRGS

DEFT routing

- Exponential function:

if $d^t(u) > d^t(v)$ then $\Gamma[h^t(u,v)] = \exp[-h^t(u,v)]$,

otherwise $\Gamma[h^t(u,v)] = 0$

- The total flow $f^t(u)$ out of node u and destined to node t is split according to:

$$f^t(u,v) = f^t(u)\ \Gamma[h^t(u,v)] / \Sigma_{(u,j)\in E} \Gamma[h^t(u,j)]$$

at&t
Your world. Delivered.

Informática
UFRGS

OSPF vs. DEFT

- Xu, Chiang, and Rexford (2007) proposed a two-stage iterative method, where each stage solves a non-linear programming problem making use of a primal-dual interior point algorithm. There method produces real-valued weights.
- In experiments described in their paper, their approach finds near-optimal solutions (i.e. near the linear programming lower bounds).

at&t
Your world. Delivered.

Informática
UFRGS

Memetic algorithm for DEFT weight setting

- Similar to memetic algorithm for OSPF weight setting
  - GA with random keys
  - fast local search
- Decoder is the only difference
  - weights are set as in MA for OSPF
  - shortest paths and gaps are determined, penalties defined, and flows computed
  - fast local search is adapted for DEFT

Implementation issues for OSPF and DEFT: dynamic updates

- The local search makes lots of unit link weight changes
- Each changed network has to be re-evaluated
  - calculate new shortest paths
  - calculate new network flow
- In both algorithms distances and flows should be updated and not recomputed from scratch

at&t
Your world. Delivered.

Informática
UFRGS

Implementation issues for OSPF and DEFT: dynamic distances

- Our MA for OSPF uses an improved version of the algorithm of Ramalingan and Reps
  - Buriol, Resende, Thorup: Speeding up dynamic shortest path algorithms, INFORMS J. Comput. 2008
- For DEFT, the algorithm is slightly faster because it only needs to update the distances and not the shortest path graphs.

Implementation issues for OSPF and DEFT: dynamic flows

- Given a change in a link flow, the flow in the graph can be updated in both protocols.
- Updates affect a larger portion of the graph in DEFT than in OSPF.
- For example: a link weight increase on a non-shortest path does not require updating in OSPF but does so in DEFT

Preliminary experimental results

- We tested 6 instances with 7 different demand matrices
- Results are averages over 3 random seeds
- Stopping criterion: 2000 generations or 500 generations without improvement
- Each run takes about 1 hour on a SGI Altrix (1.6Ghz Itanium 2 processor)

at&t
Your world. Delivered.

Informática
UFRGS

Preliminary experimental results

- DEFT reduces network congestion when compared to OSPF
  - NLP as well as MA
  - Running times are about the same for both OSPF & DEFT MA implementations
- Optimality gap is small for both (MA, NLP)
- Some instances permit large improvements (hier50b, hier100a), others almost none (waxman)

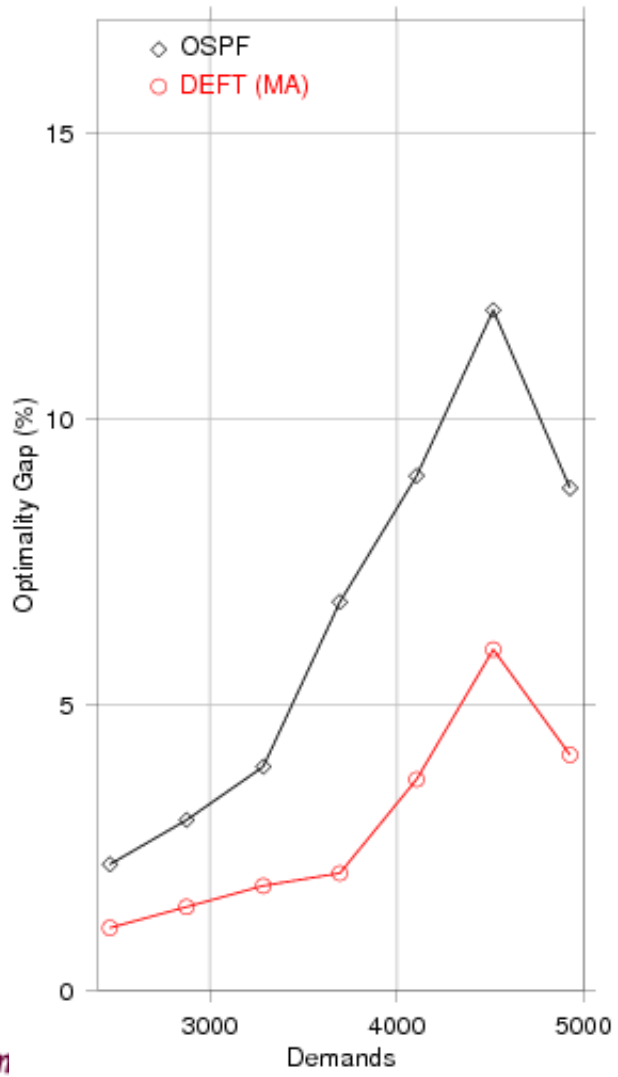
at&t
Your world. Delivered.

Informática
UFRGS

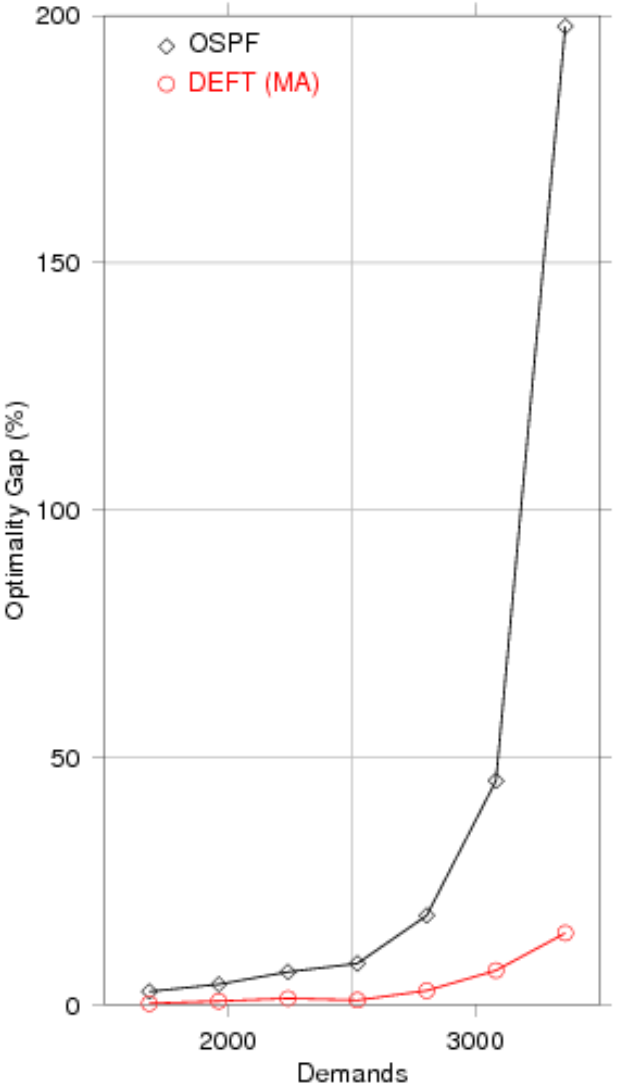
OSPF vs. DEFT

Two level hierarchy with 50 nodes

hier50a
148 links



Hier50b
212 links



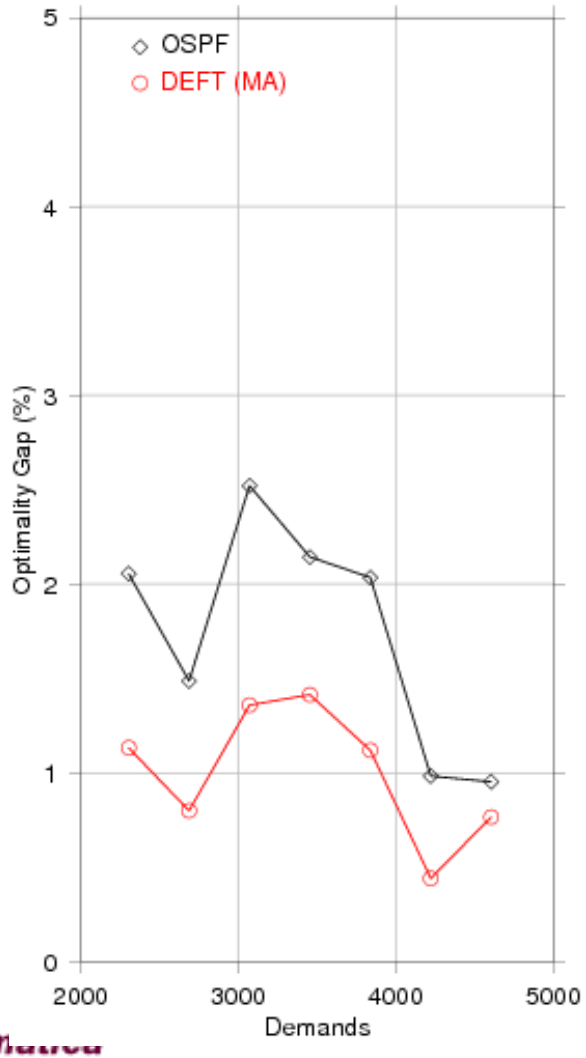
at&t
Your world. Delivered.

Inform
UFRGS

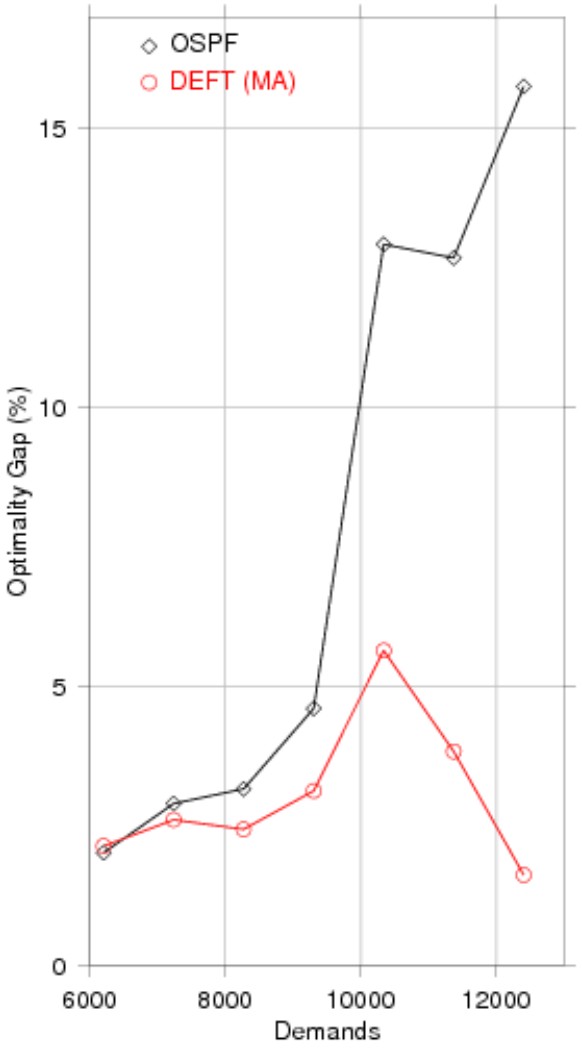
OSPF vs. DEFT

Two level hierarchy with 100 nodes

Hier100
280 links



Hier100a
360 links

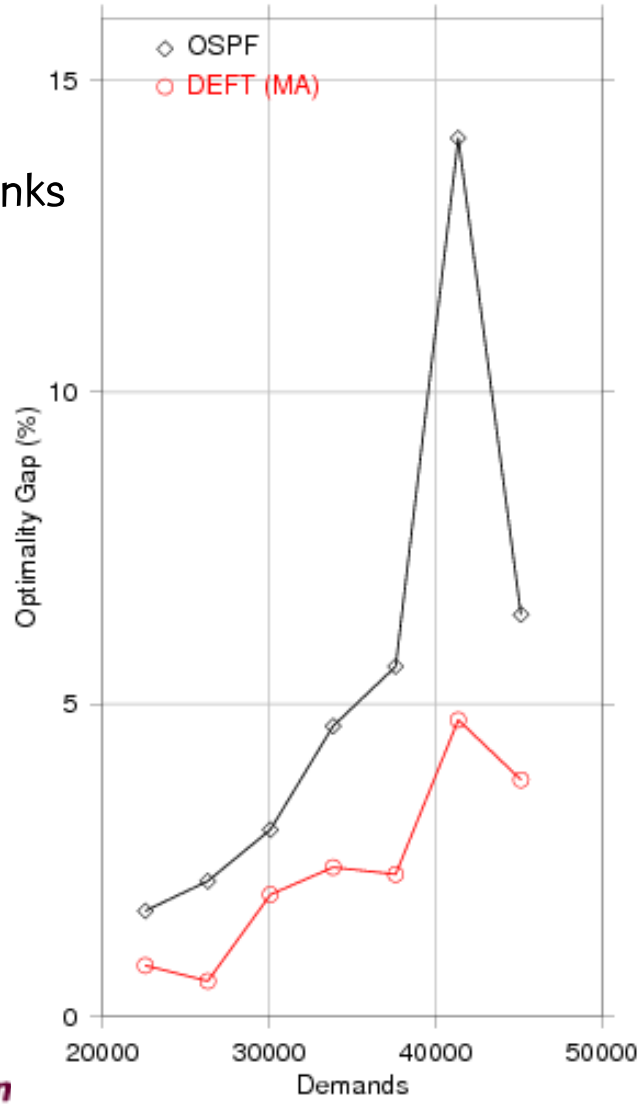


at&t
Your world. Delivered.

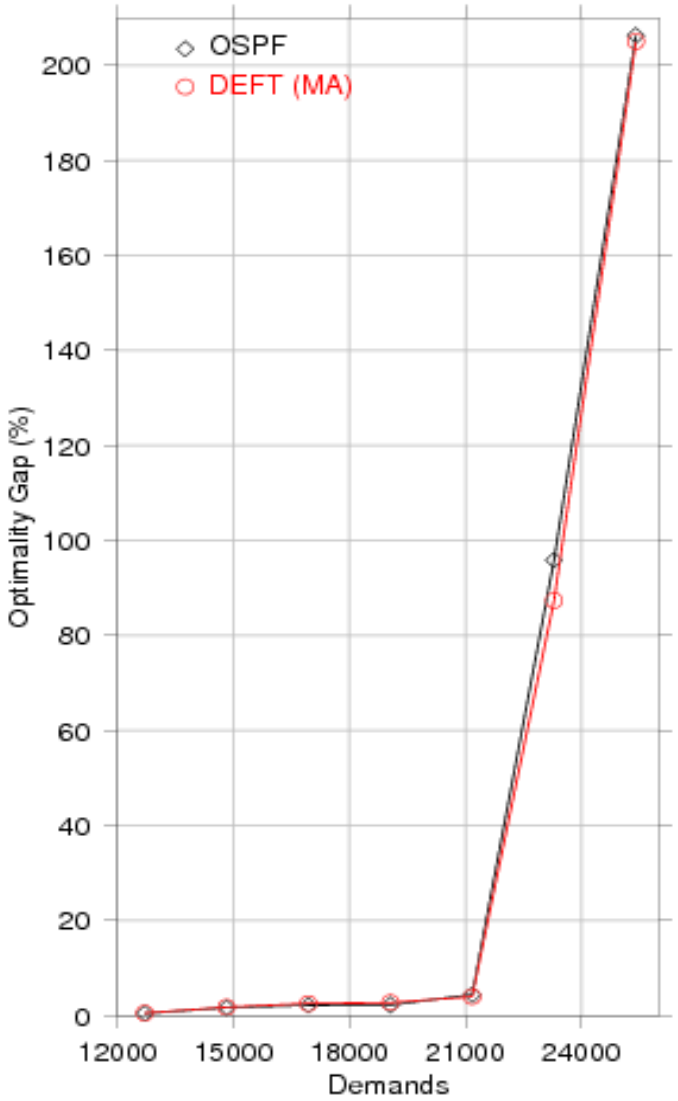
UFRGS

OSPF vs. DEFT

AT&T
90 nodes, 274 links



Waxman
50 nodes,
169 links

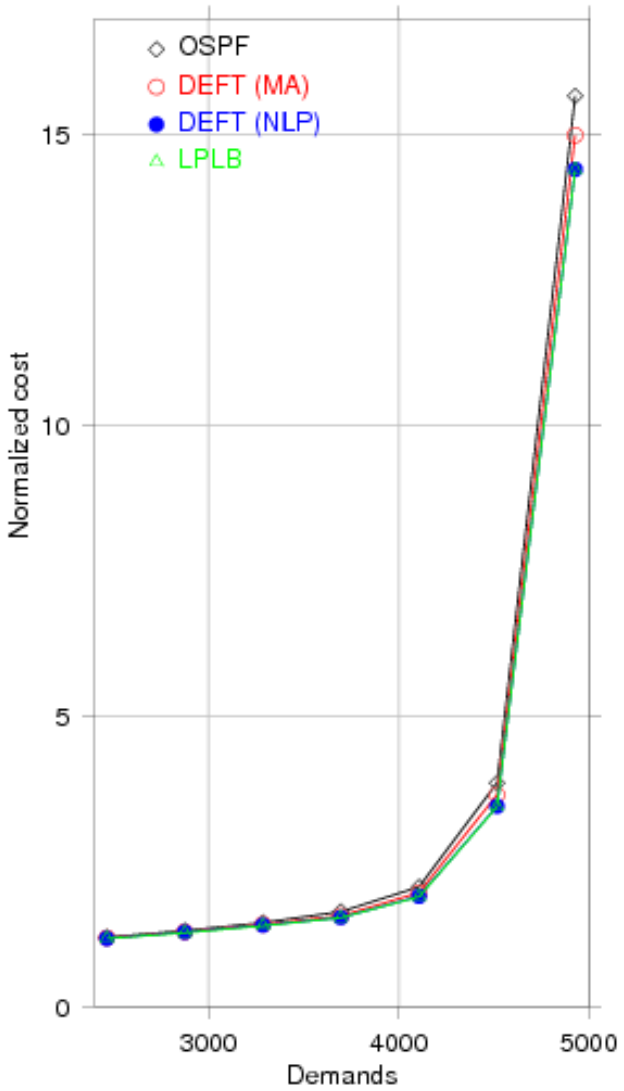


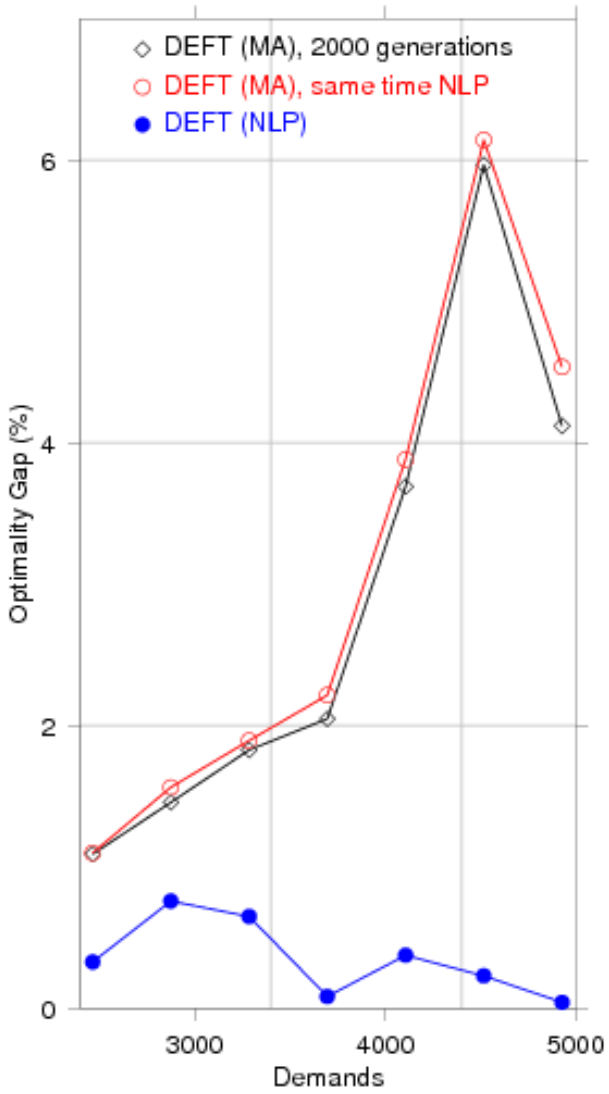
at&t
Your world. Delivered.

Inform UFRGS

DEFT
Comparison of NLP and MA approaches

Hier50a
50 nodes, 148 links





at&t
Your world. Delivered.

Informática
UFRGS

Conclusion

- DEFT routing archives better results than OSPF routing
- Further experiments are necessary to conclude if it is worthwhile to switch to DEFT
- Memetic algorithm is a flexible tool
  - permits survivable network design (e.g. multiplicities)
  - permits parallelization (e.g. island methods)

at&t
Your world. Delivered.

Informática
UFRGS

Future work

- Compare the network delay of DEFT and OSPF
- Explore DEFT with real weights
  - Better local search using line search algorithms
  - Continuous optimization methods, e.g. continuous GRASP
- Parallelize the memetic algorithm
- Implement survivable network design using DEFT

at&t
Your world. Delivered.

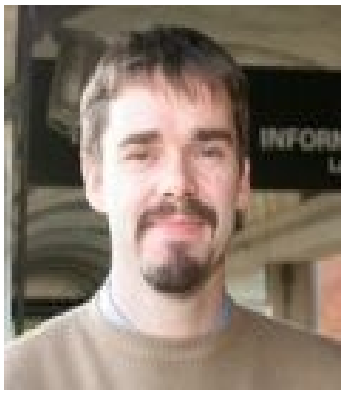
Informática
UFRGS

Thanks to...

Dahai Xu, Mung Chiang, and Jennifer Rexford for sharing their implementation and insights

Coauthors

Luciana Buriol
Universidade Federal do
Rio Grande do Sul, Brazil



Marcus Ritt,
Universidade Federal do
Rio Grande do Sul, Brazil

Roger Reis
Universidade Federal do
Rio Grande do Sul, Brazil

at&t
Your world. Delivered.

Informática
UFRGS

The End

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

http://mauricioresende.com