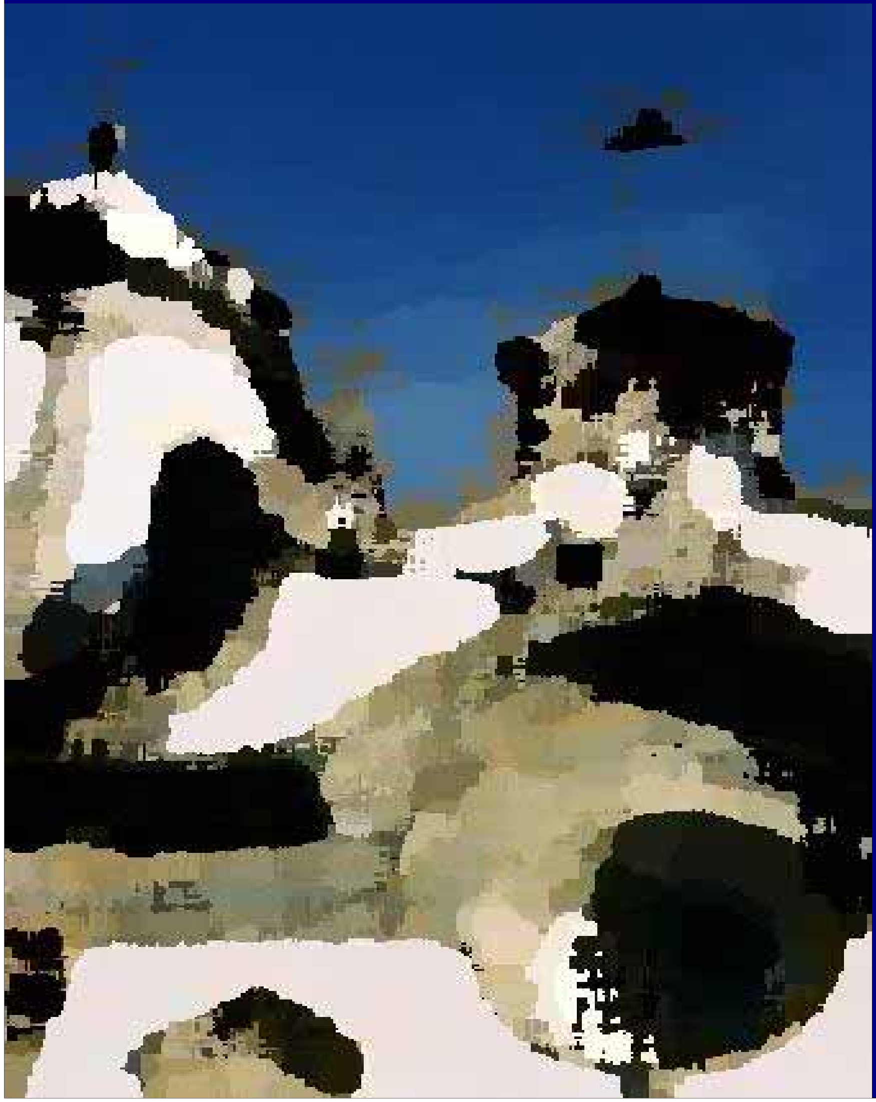
Otimização Combinatória nas Telecomunicações

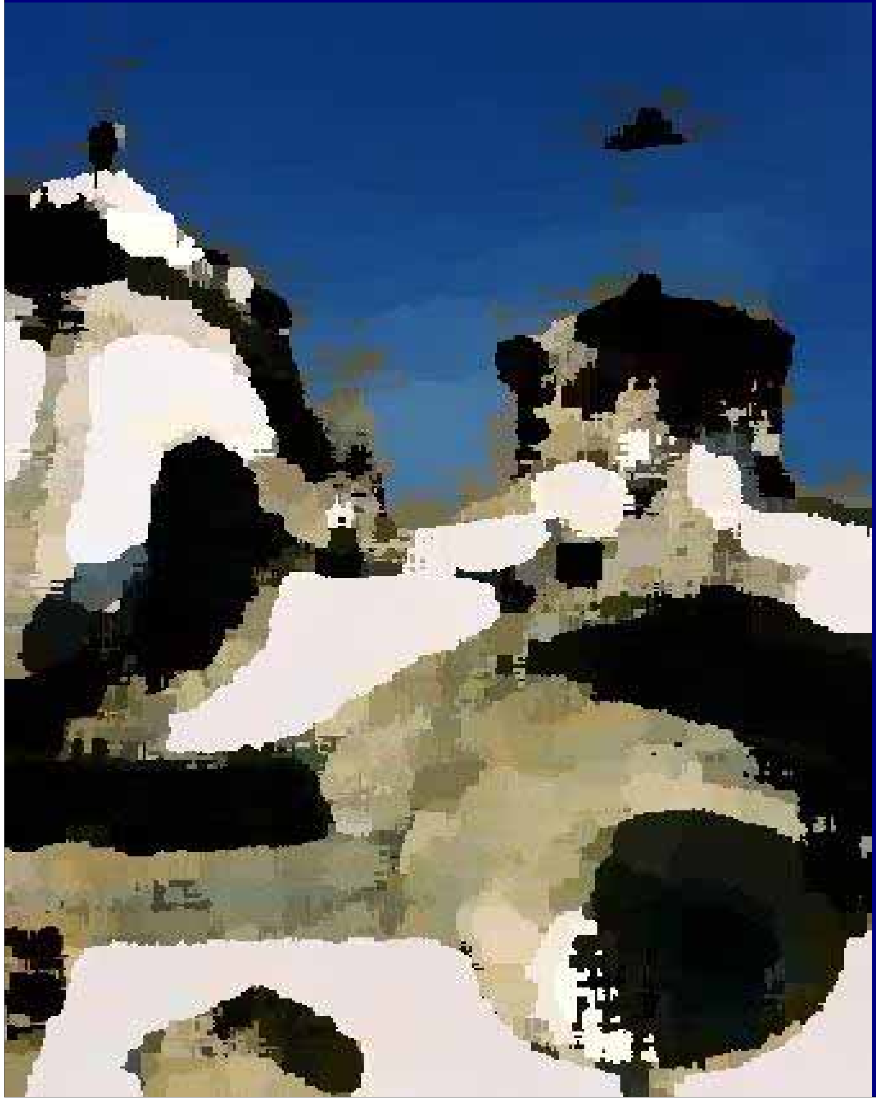


Mauricio G. C. Resende

AT&T Labs Research
Florham Park, New Jersey
mgcr@research.att.com
www.research.att.com/~mgcr

XXXVI SBPO
São João del Rei, MG
23 a 26 de novembro de 2004

Combinatorial Optimization in Telecommunications



Mauricio G. C. Resende

AT&T Labs Research
Florham Park, New Jersey
mgcr@research.att.com
www.research.att.com/~mgcr

XXXVI SBPO
São João del Rei, Brazil
November 23 to 26, 2004

Summary of talk

- Network migration
- Modem pool placement for Internet service provider
- Local access network design
- Traffic routing on a virtual private network
- Internet traffic engineering
- Survivable IP network design

Summary of talk

- Network migration
- Modem pool placement for Internet service provider
- Local access network design
- Traffic routing on a virtual private network
- Internet traffic engineering
- Survivable IP network design

Summary of talk

- Network migration
- Modem pool placement for Internet service provider
- Local access network design
- Traffic routing on a virtual private network
- Internet traffic engineering
- Survivable IP network design

Summary of talk

- Network migration
- Modem pool placement for Internet service provider
- Local access network design
- Traffic routing on a virtual private network
- Internet traffic engineering
- Survivable IP network design

Summary of talk

- Network migration
- Modem pool placement for Internet service provider
- Local access network design
- Traffic routing on a virtual private network
- Internet traffic engineering
- Survivable IP network design

Summary of talk

- Network migration
- Modem pool placement for Internet service provider
- Local access network design
- Traffic routing on a virtual private network
- Internet traffic engineering
- Survivable IP network design

Application 1:

Network migration scheduling

Network migration scheduling

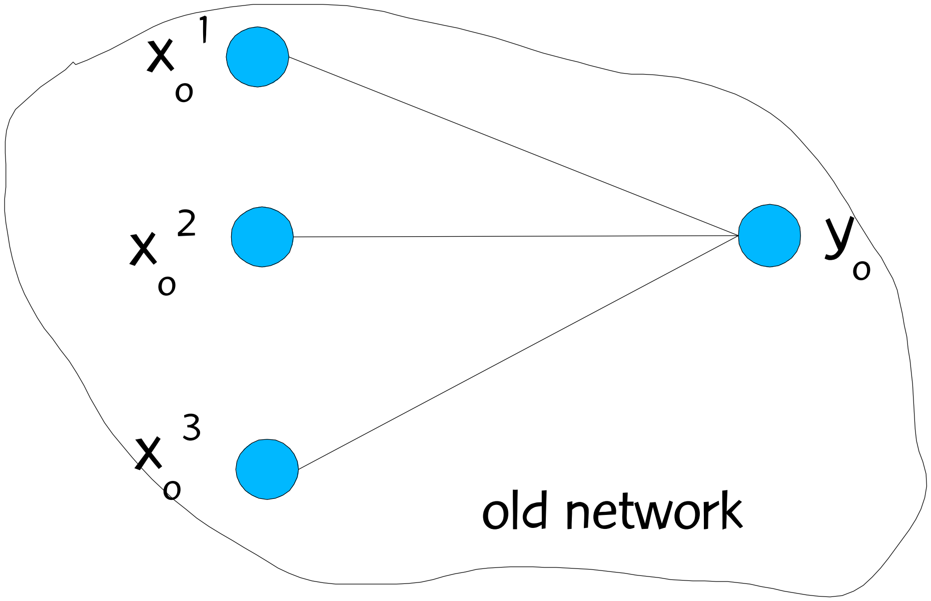
- Inter-nodal traffic from an outdated network is migrated to a new network.
- All traffic originating or terminating at a given node in the outdated network is moved to a specific node in the new network.
- Routing is predetermined in both networks and therefore capacities are known.

Network migration scheduling

- Traffic between nodes in the same network is routed in that network.
- Suppose node y_o in the old network is migrated to node y_n in the new network.
- Let link (x_o , y_o) have capacity c_o.
- Traffic from x_o to y_n must use a temporary link (x_o , y_n) with capacity c_o.

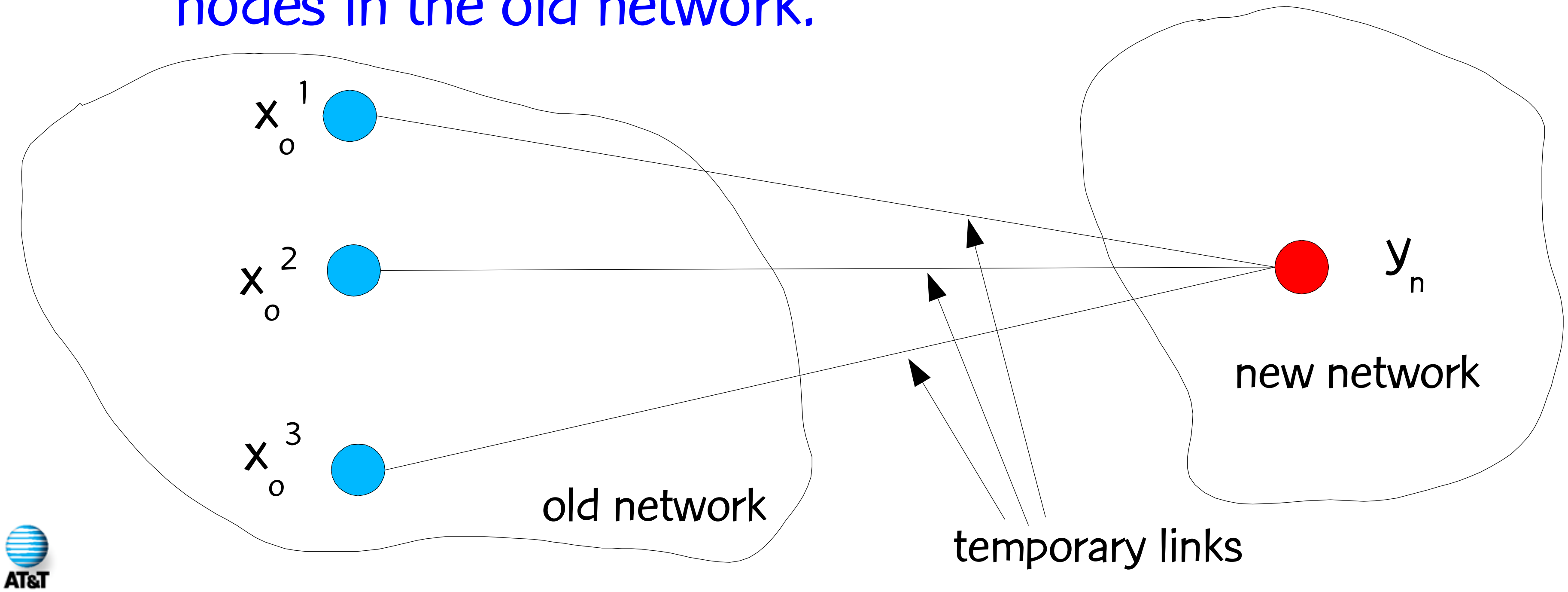
Network migration scheduling

- When node y_o is migrated to y_n in the new network, one or more temporary links may have to be used, since node y_o may be adjacent to one or more still-active nodes in the old network.



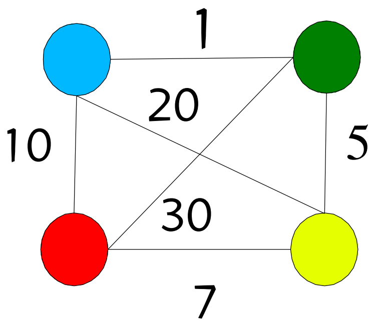
Network migration scheduling

- When node y_o is migrated to y_n in the new network, one or more temporary links may have to be used, since node y_o may be adjacent to one or more still-active nodes in the old network.



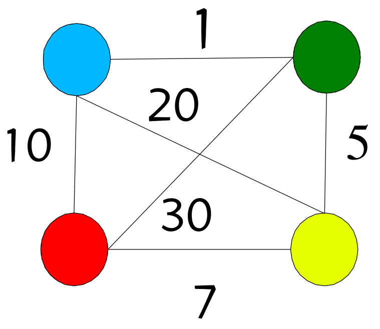
Network migration scheduling problem

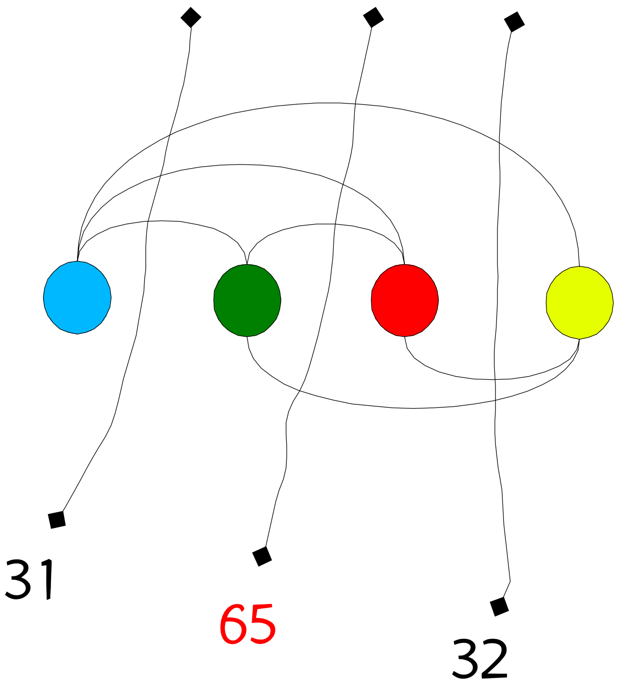
- Find a migration ordering of the vertices such that the maximum sum of the capacities of the temporary links is minimized.



Network migration scheduling problem

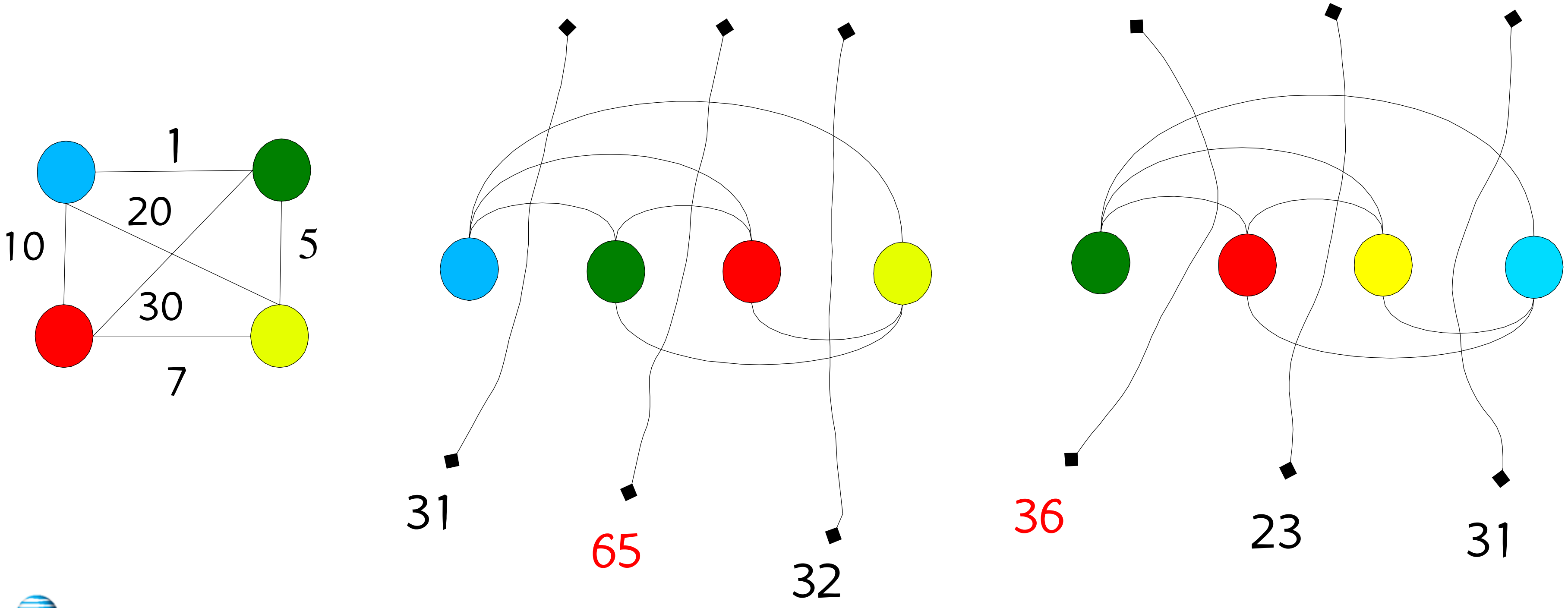
- Find a migration ordering of the vertices such that the maximum sum of the the capacities of the temporary links is minimized.





Network migration scheduling problem

- Find a migration ordering of the vertices such that the maximum sum of the the capacities of the temporary links is minimized.



GRASP for MCLA

- Given $G = (V,E)$ and weights w_{uv} on links (u,v). Find permutation of nodes $\pi_1, \pi_2, \ldots, \pi_n$ defining the schedule.
- Suppose nodes $\pi_1, \pi_2, \ldots, \pi_{k-1}$ have been already scheduled and let $\Omega = V \setminus \{ \pi_1, \pi_2, \ldots, \pi_{k-1} \}$ be the set of yet to be scheduled nodes.

GRASP for MCLA

- Let f(u) be the sum of link weights from node u to all nodes in Ω \ {u}: choose u with small f(u)
- Likewise, let b(u) be the sum of link weights from u to all nodes in V \ Ω \ {u}: choose u with largest b(u)
- Greedy choice: choose u with smallest f(u) – b(u)
- Greedy randomized choice: choose u from set of nodes with small f(u) – b(u) value.
- Local search: swap order of node pairs (observe only cuts between nodes are potentially affected by swap)

GRASP for MCLA

```
repeat {
        $\pi$ = GreedyRandomizedConstruction(■);
        $\pi$ = Local Search($\pi$);
        save $\pi$ as $\pi^*$ if best so far;
}
return $\pi^*$;
```

Path-relinking

- Path-relinking:
  - Intensification strategy exploring trajectories connecting elite solutions: Glover (1996)
  - Originally proposed in the context of tabu search and scatter search.
  - Paths in the solution space leading to other elite solutions are explored in the search for better solutions:
    - selection of moves that introduce attributes of the guiding solution into the current solution

Path-relinking

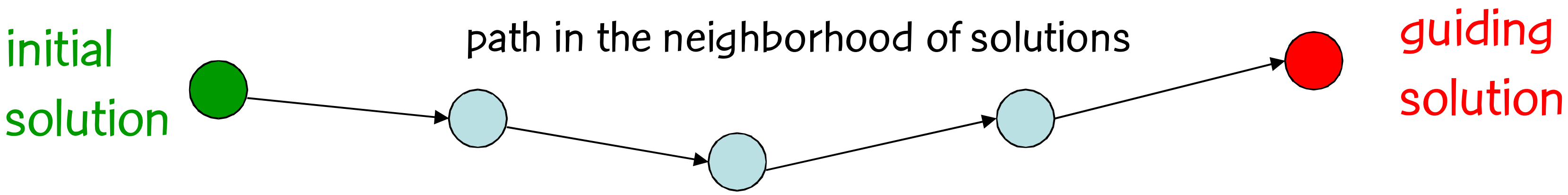
- Path-relinking:
  - Intensification strategy exploring trajectories connecting elite solutions: Glover (1996)
  - Originally proposed in the context of tabu search and scatter search.
  - Paths in the solution space leading to other elite solutions are explored in the search for better solutions:
    - selection of moves that introduce attributes of the guiding solution into the current solution

Path-relinking

- Path-relinking:
  - Intensification strategy exploring trajectories connecting elite solutions: Glover (1996)
  - Originally proposed in the context of tabu search and scatter search.
  - Paths in the solution space leading to other elite solutions are explored in the search for better solutions:
    - selection of moves that introduce attributes of the guiding solution into the current solution

Path-relinking

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

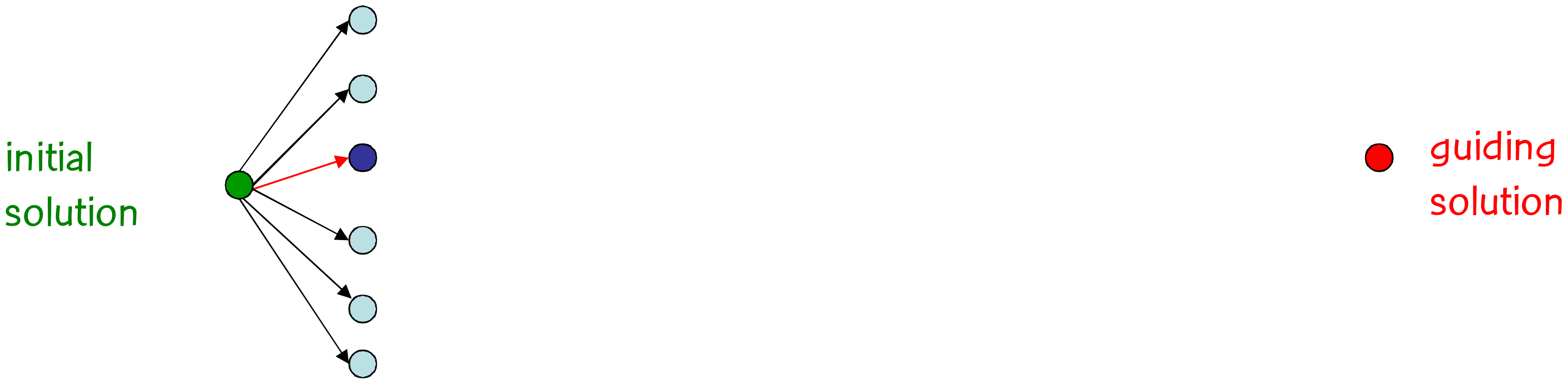


Path-relinking

- 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:

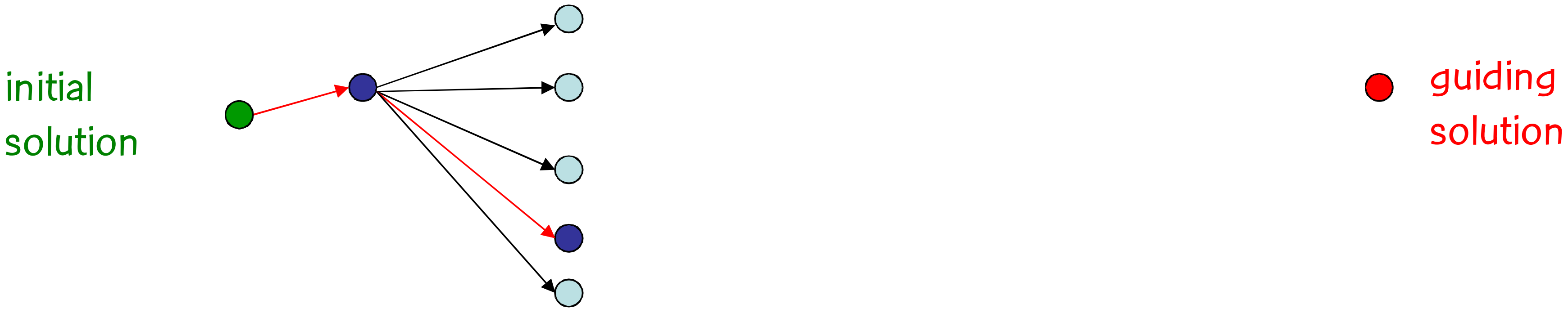
Path-relinking

- 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:

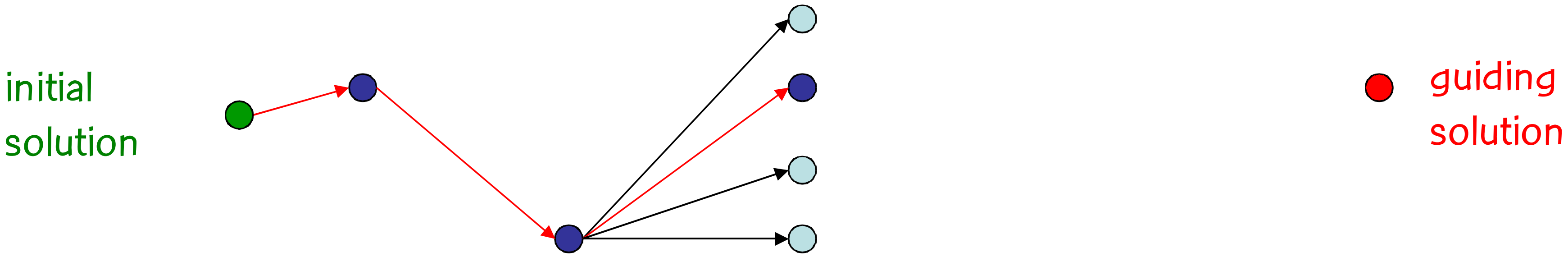


initial solution

guiding solution

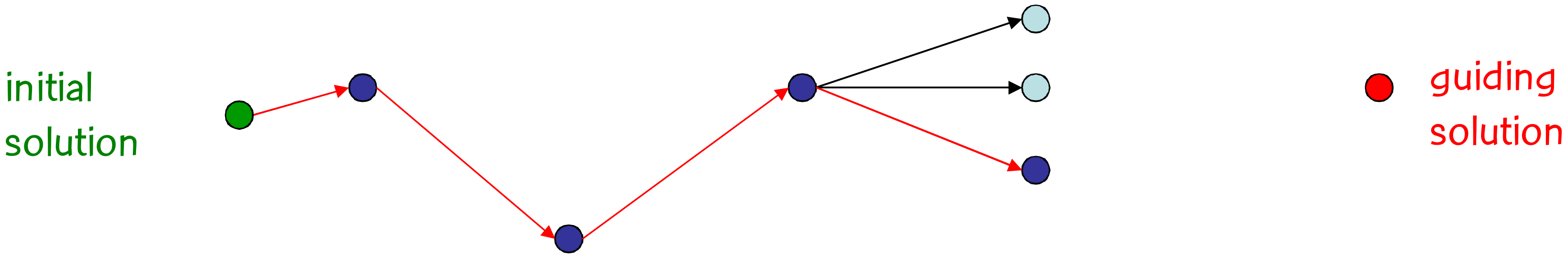
Path-relinking

- 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:



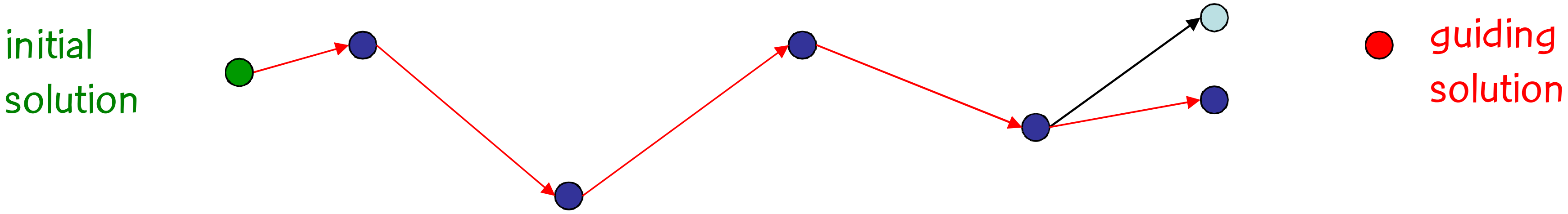
Path-relinking

- 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:



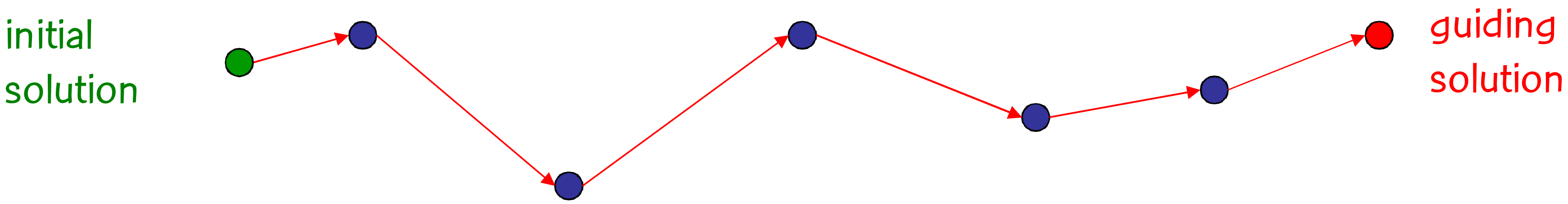
Path-relinking

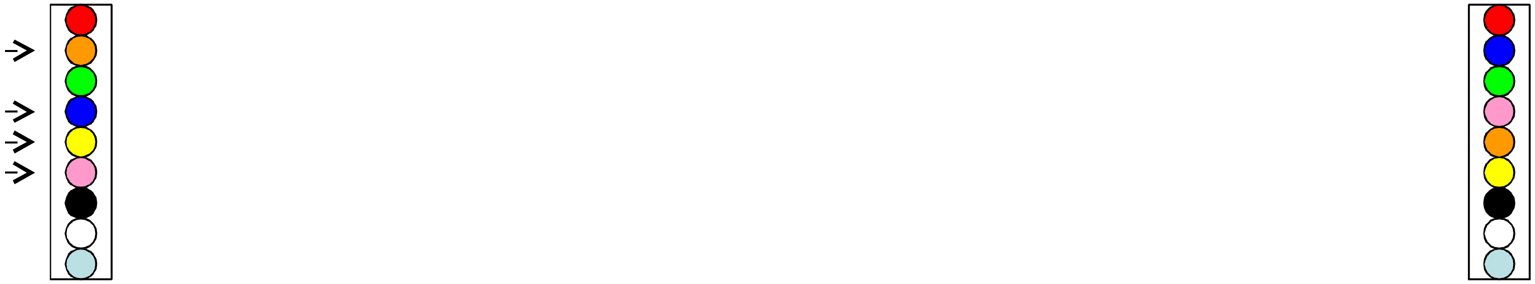
- Path is generated by selecting moves that introduce in the initial solution attributes of the guiding solution.
- At each step, all moves that incorporate attributes of the guiding solution are evaluated and the best move is selected:



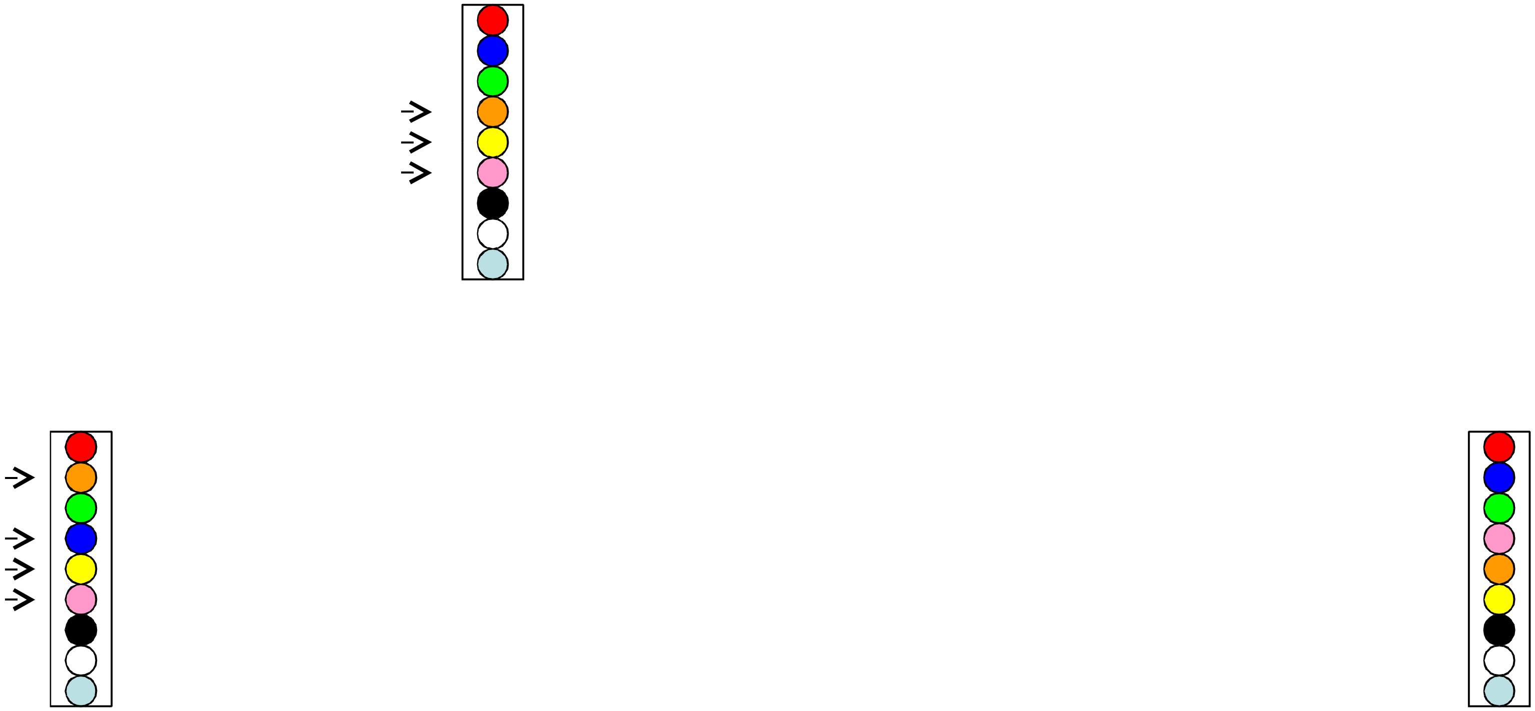
Path-relinking

- 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:

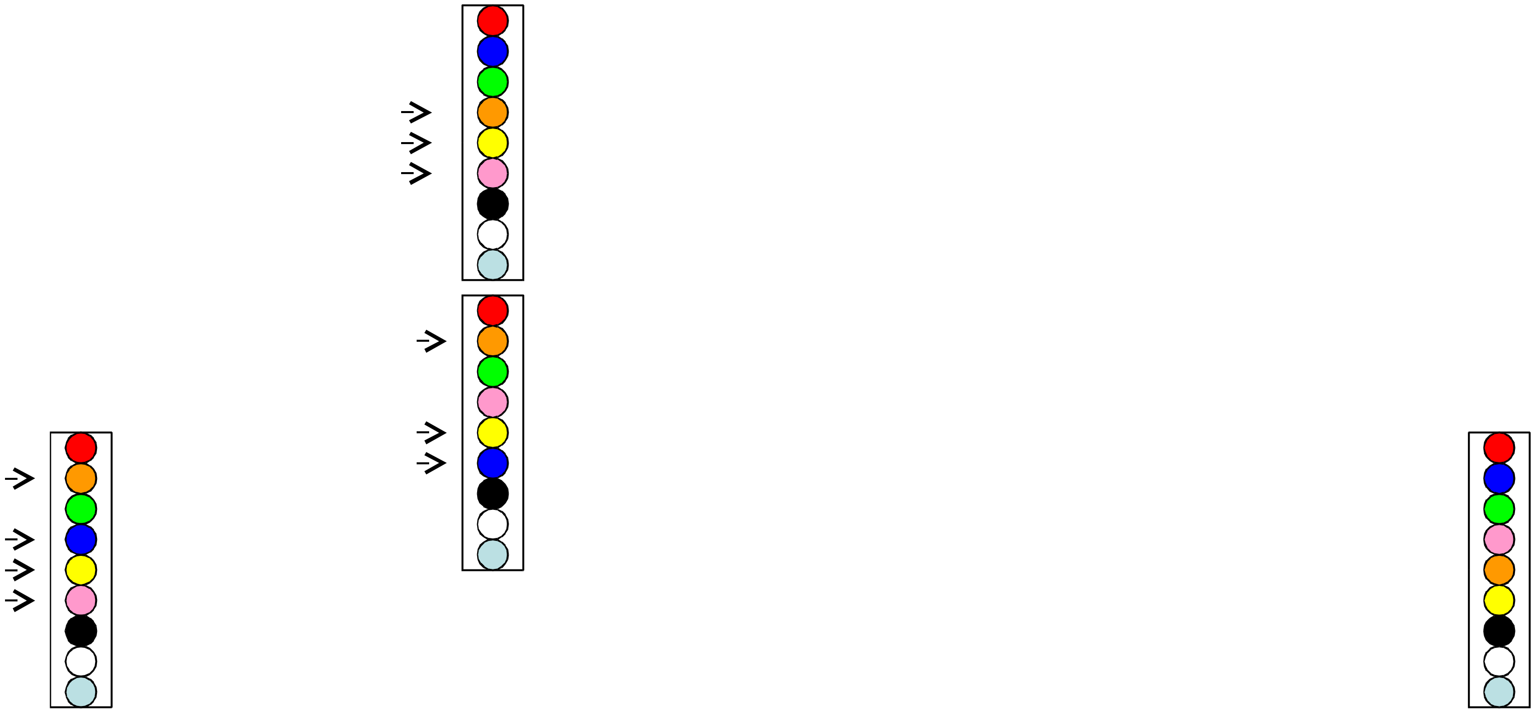




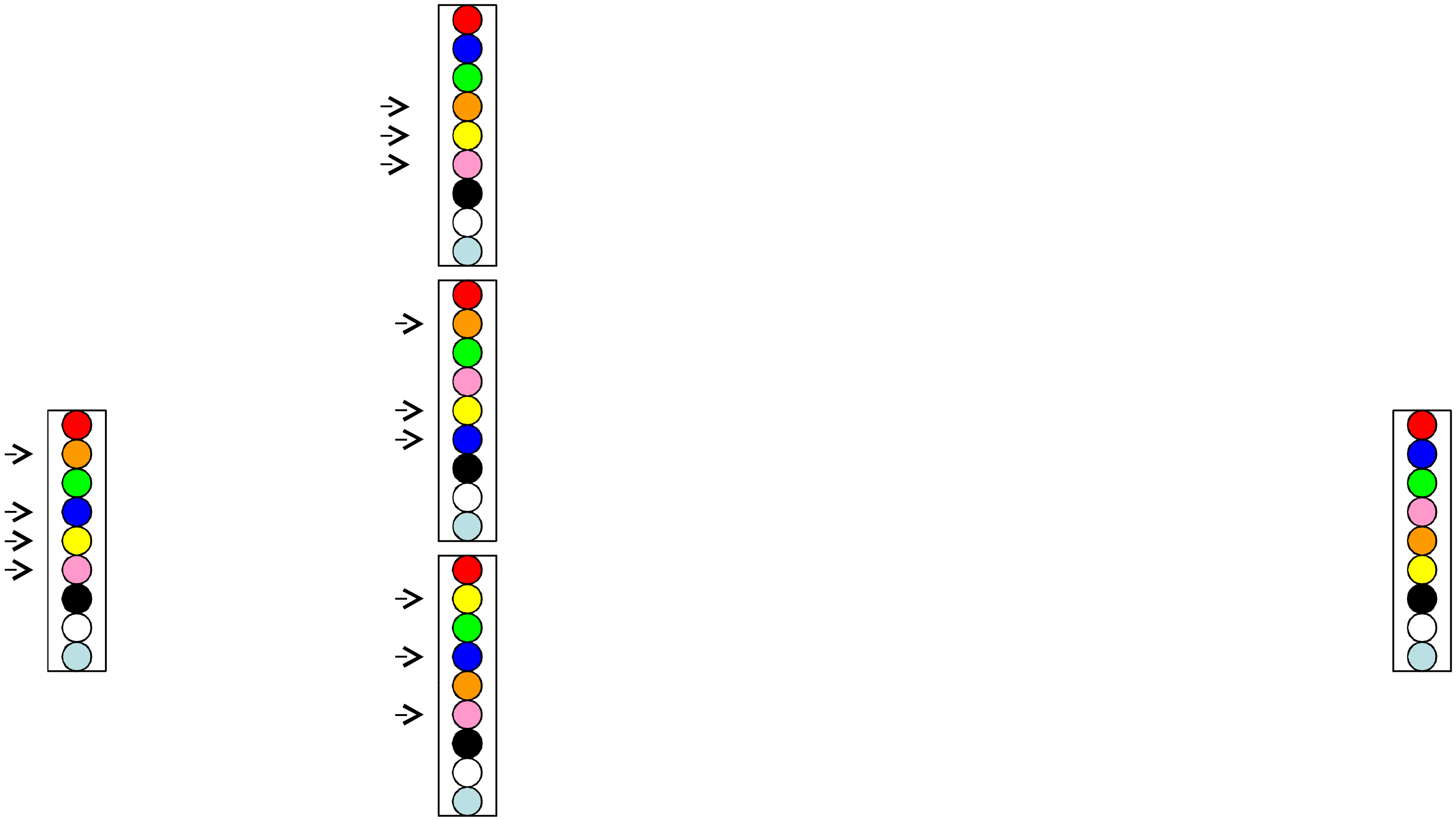
Path-relinking for the MCLA problem



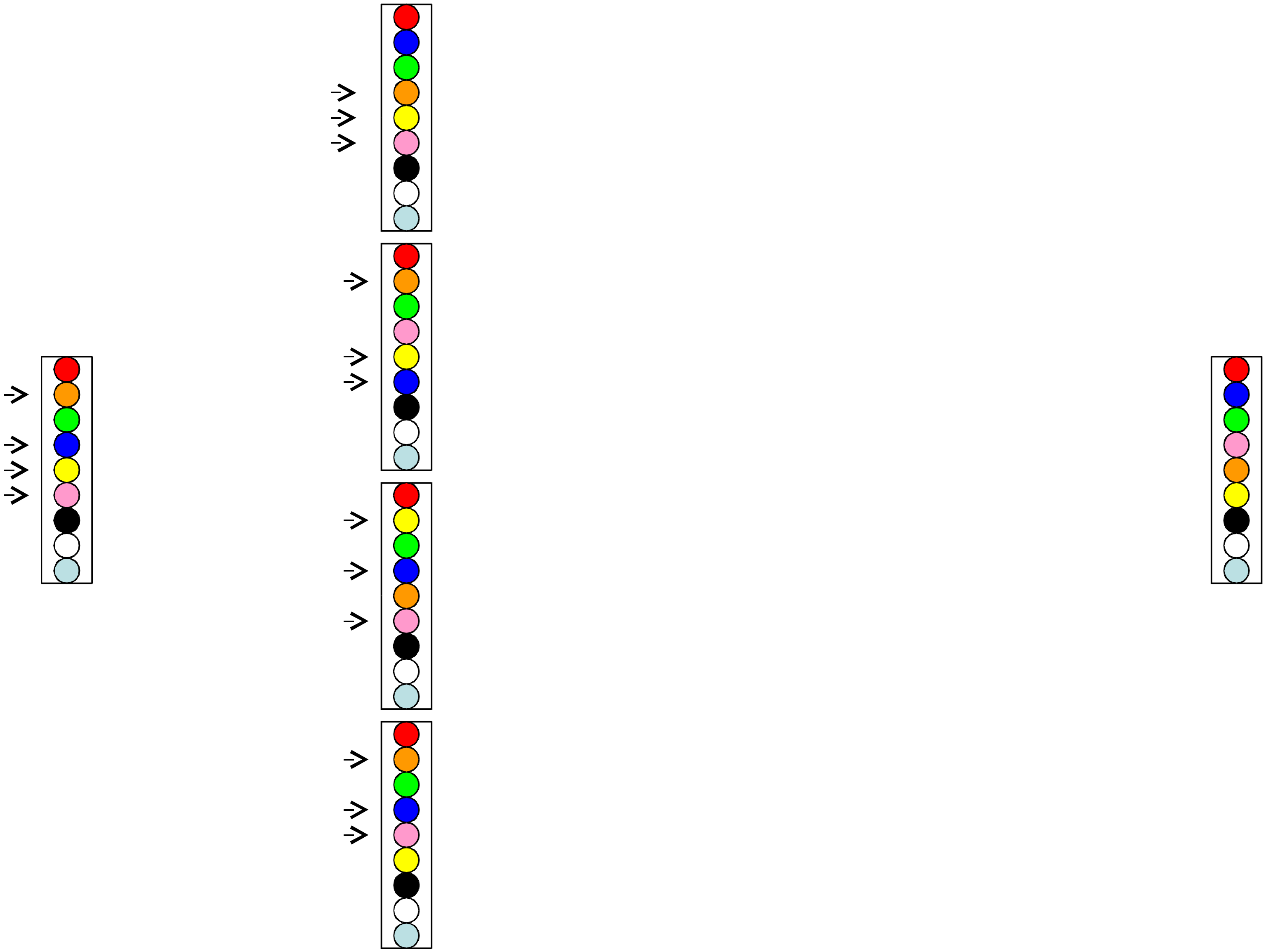
Path-relinking for the MCLA problem



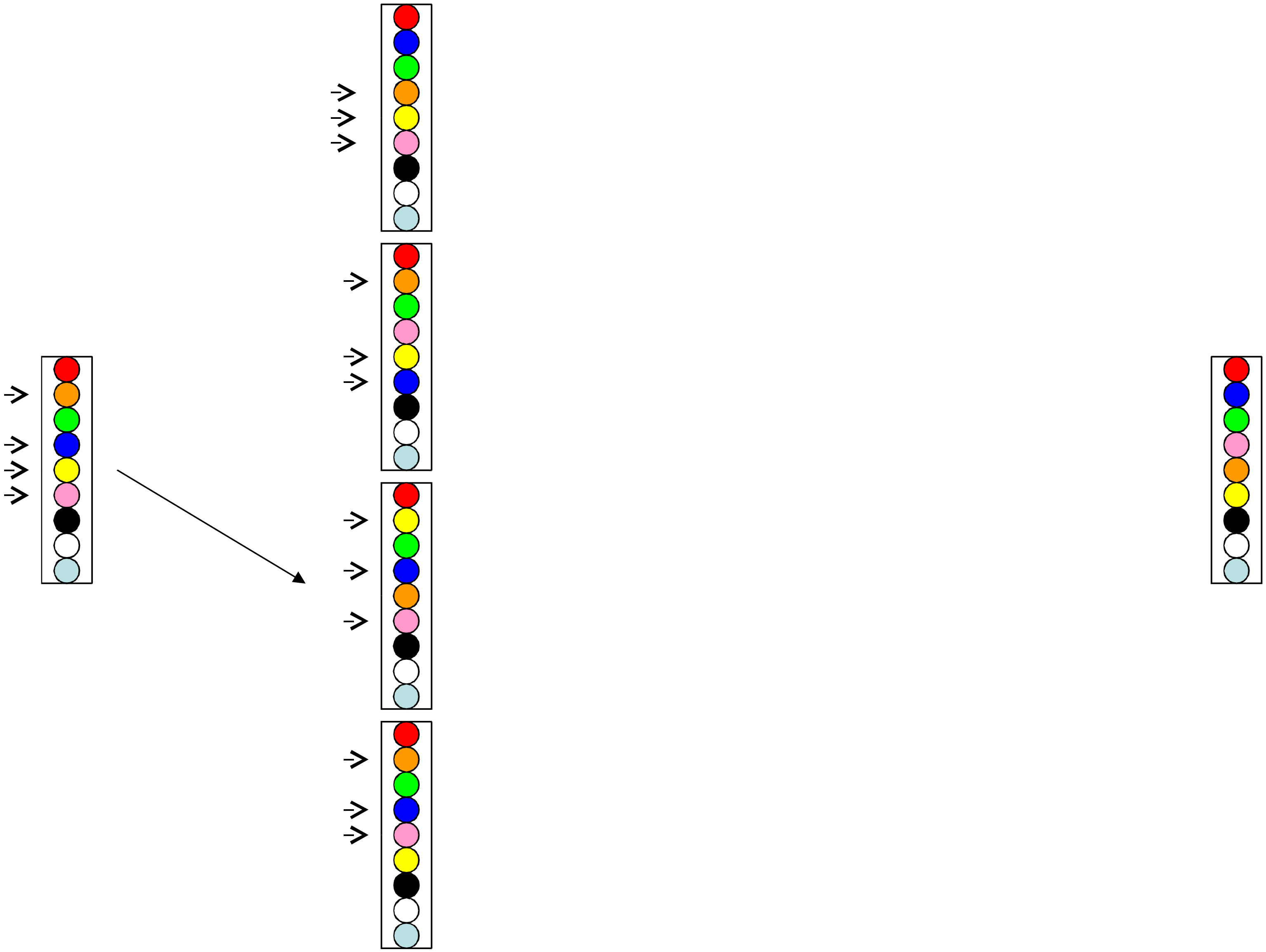
Path-relinking for the MCLA problem



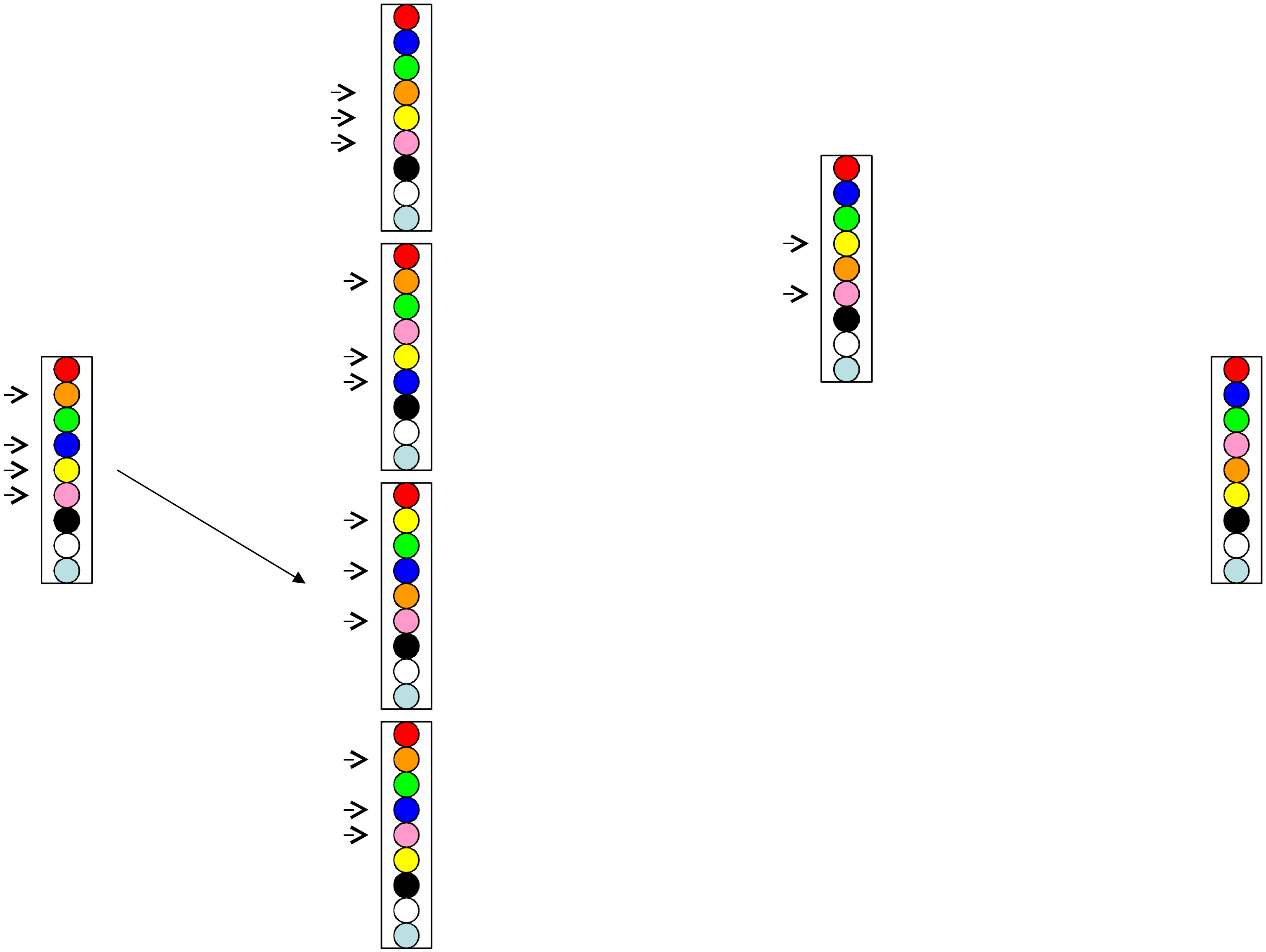
Path-relinking for the MCLA problem



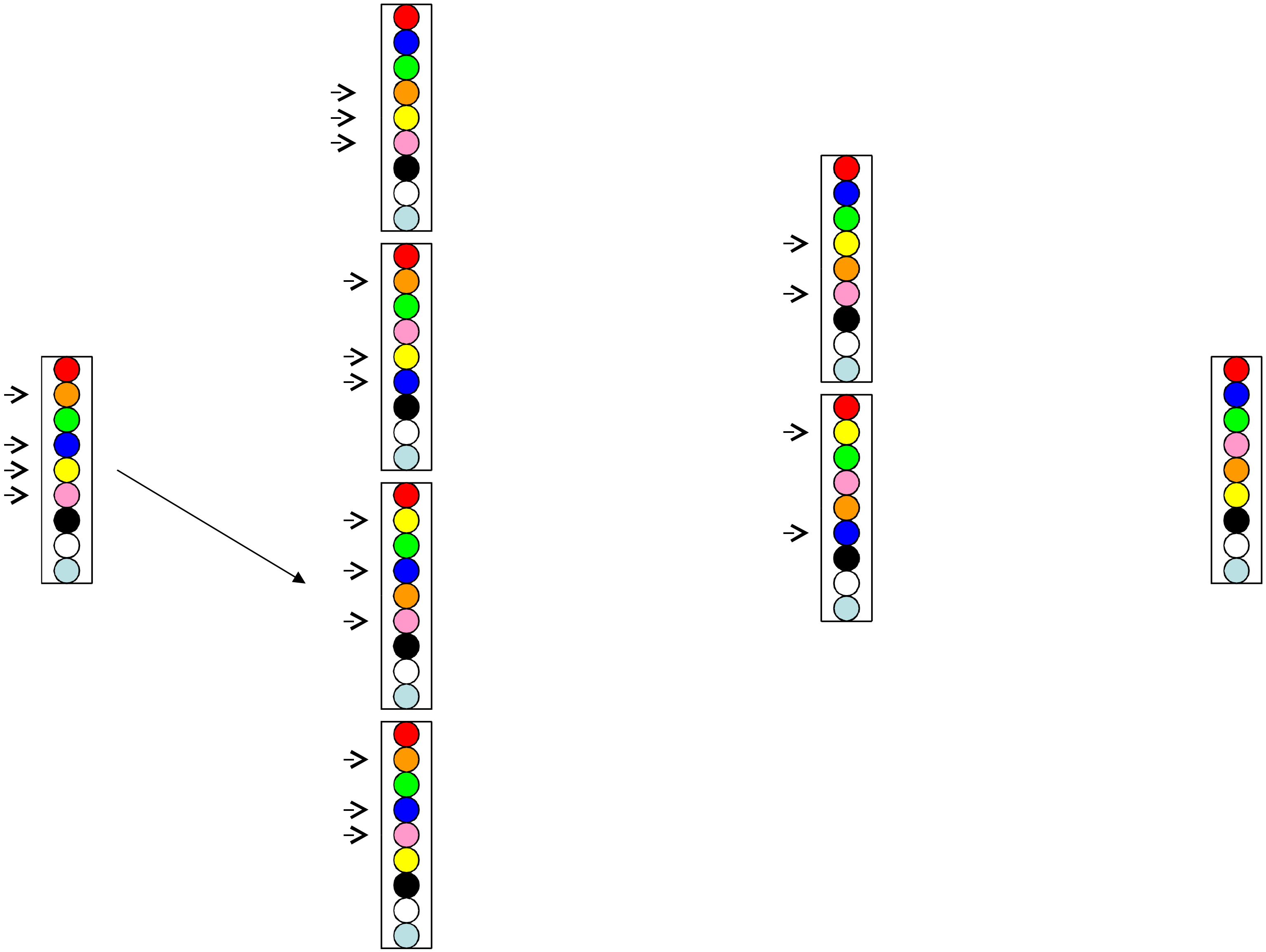
Path-relinking for the MCLA problem



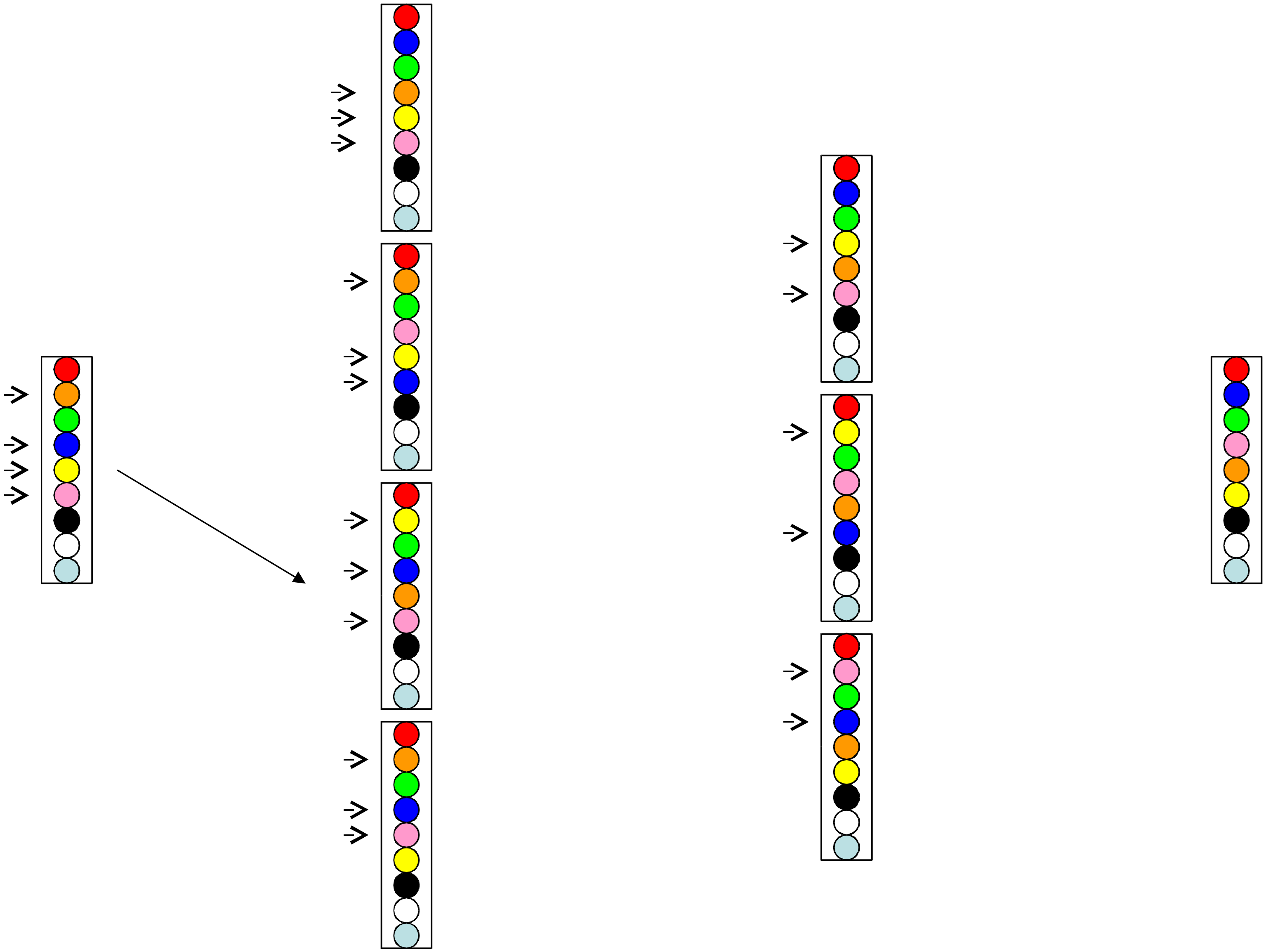
Path-relinking for the MCLA problem



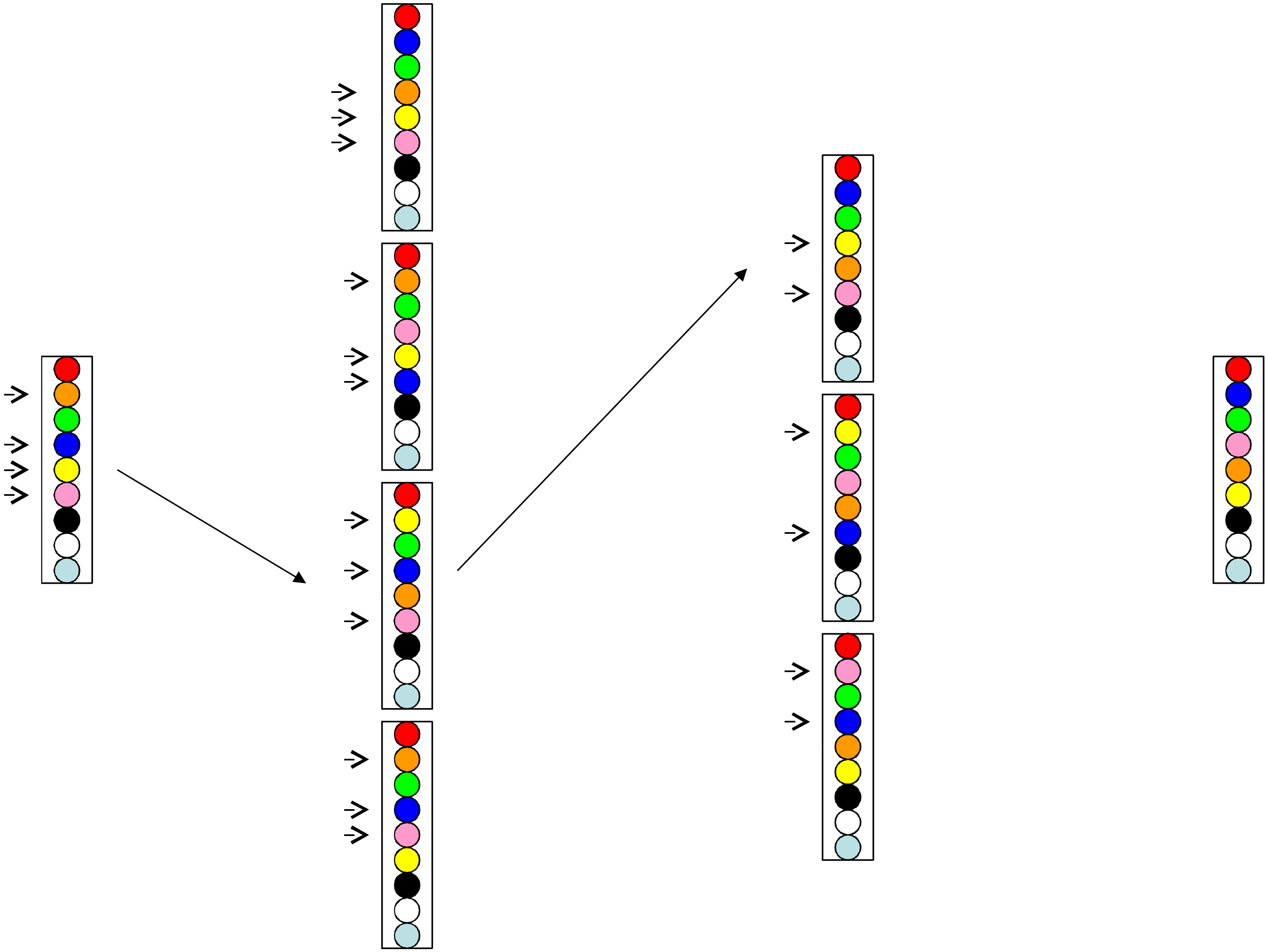
Path-relinking for the MCLA problem



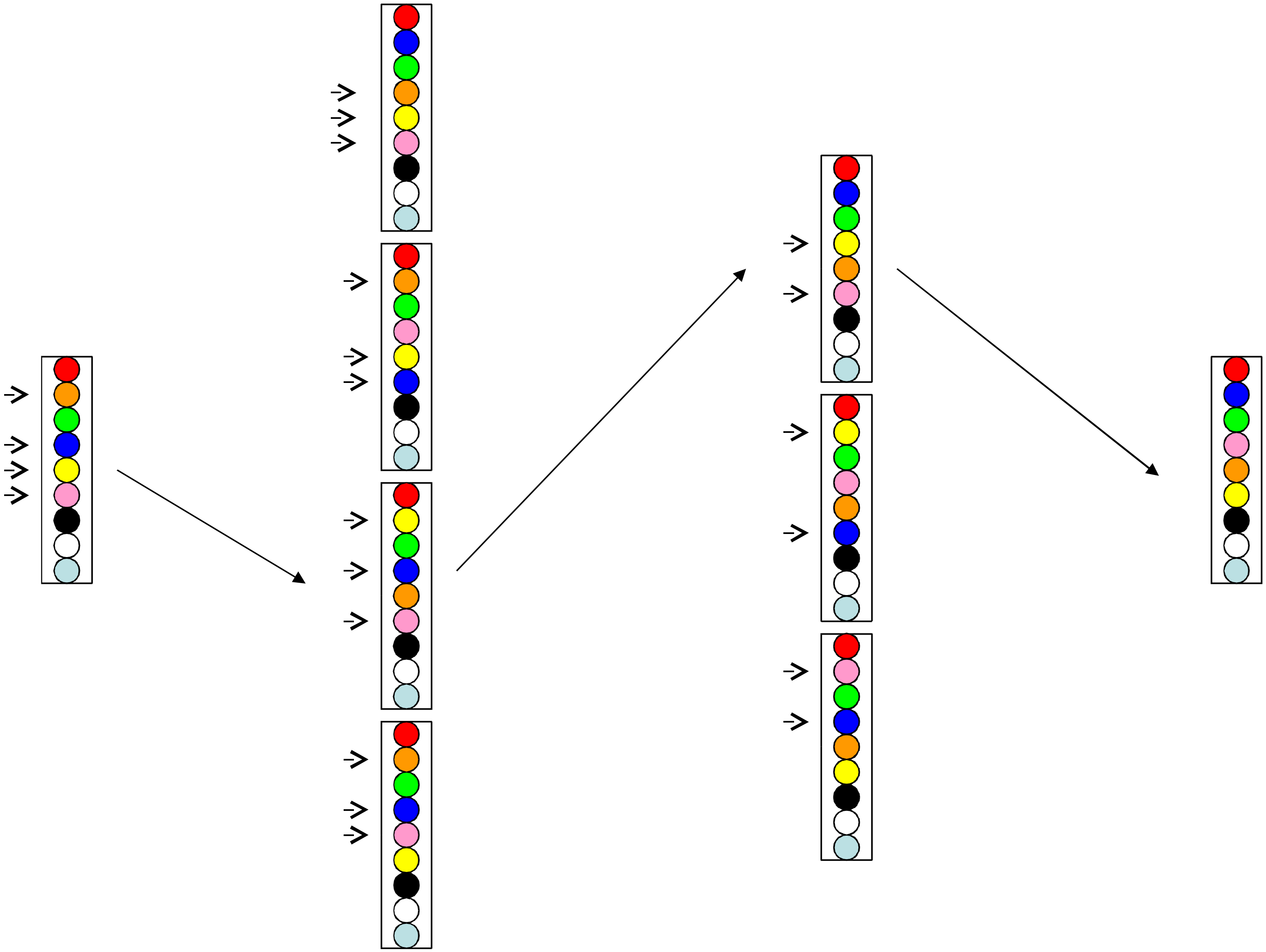
Path-relinking for the MCLA problem



Path-relinking for the MCLA problem



Path-relinking for the MCLA problem

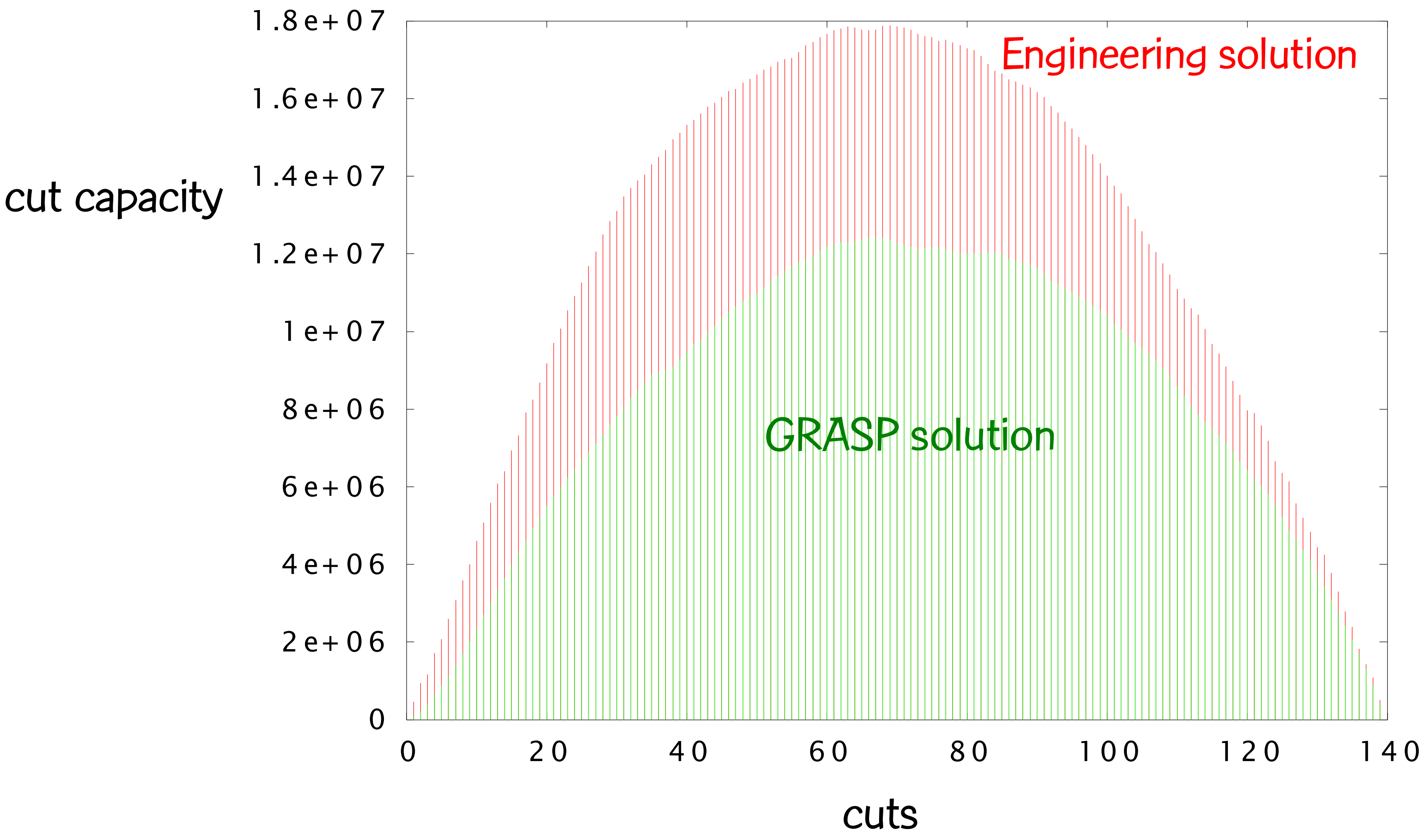


Path-relinking for the MCLA problem

A real-world migration example: 140 nodes, 9730 links

Engineering solution: 1.78×10^7

GRASP solution : 1.23×10^7 (31% reduction)



Application 2:

Modem pool placement for Internet service provider

Modem pool placement for Internet service provider

- Worldnet: AT&T's Internet Service Provider
- Dial-up: hundreds of points of presence (PoPs)
  - Telephone numbers customers must call when making an Internet connection.
- Current footprint:
  - 1305 PoPs;
  - 77.66% of the telephone numbers in the U.S. can make local calls to Worldnet.

Footprint Optimization

- In general: more PoPs, better coverage.
- For a fixed coverage, we don't want more PoPs than necessary.
- Not all PoPs are the same:
  - Each has an associated network cost:
    - Hourly rate paid by Worldnet to network company.
    - Between $0.04 and $0.14 in the continental US.
    - Up to $0.42 in Hawaii and Alaska.
  - No setup cost.

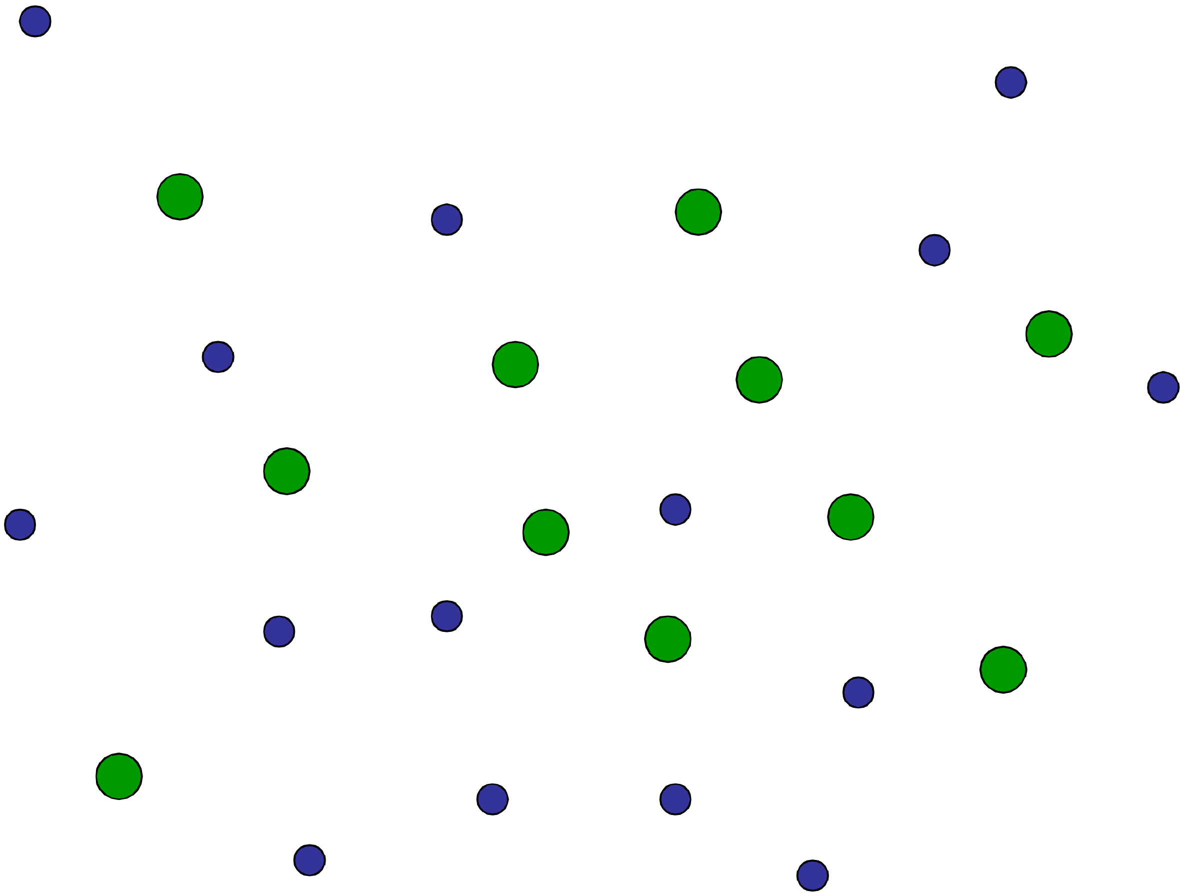
Worldnet

- When is a call local?
  - Not simply “within same area code”.
  - Telephone system divided into exchanges:
    - Area code + first three digits (973360, for example).
- Each PoP has a coordinate.
- We know which exchanges can make local calls to each coordinate (the coverage).
  - Just a big table;
  - 69,534 exchanges covered by current footprint.
- Goal: keep only cheaper PoPs, preserve coverage.

Footprint Optimization

- Further improvement:
  - 335 additional coordinates could be eliminated:
    - Only 700 PoPs left;
    - New footprint covers all exchanges currently covered;
    - No exchange has to make a more expensive call.
- How did we do it?
  - We solved this as the p-median problem.

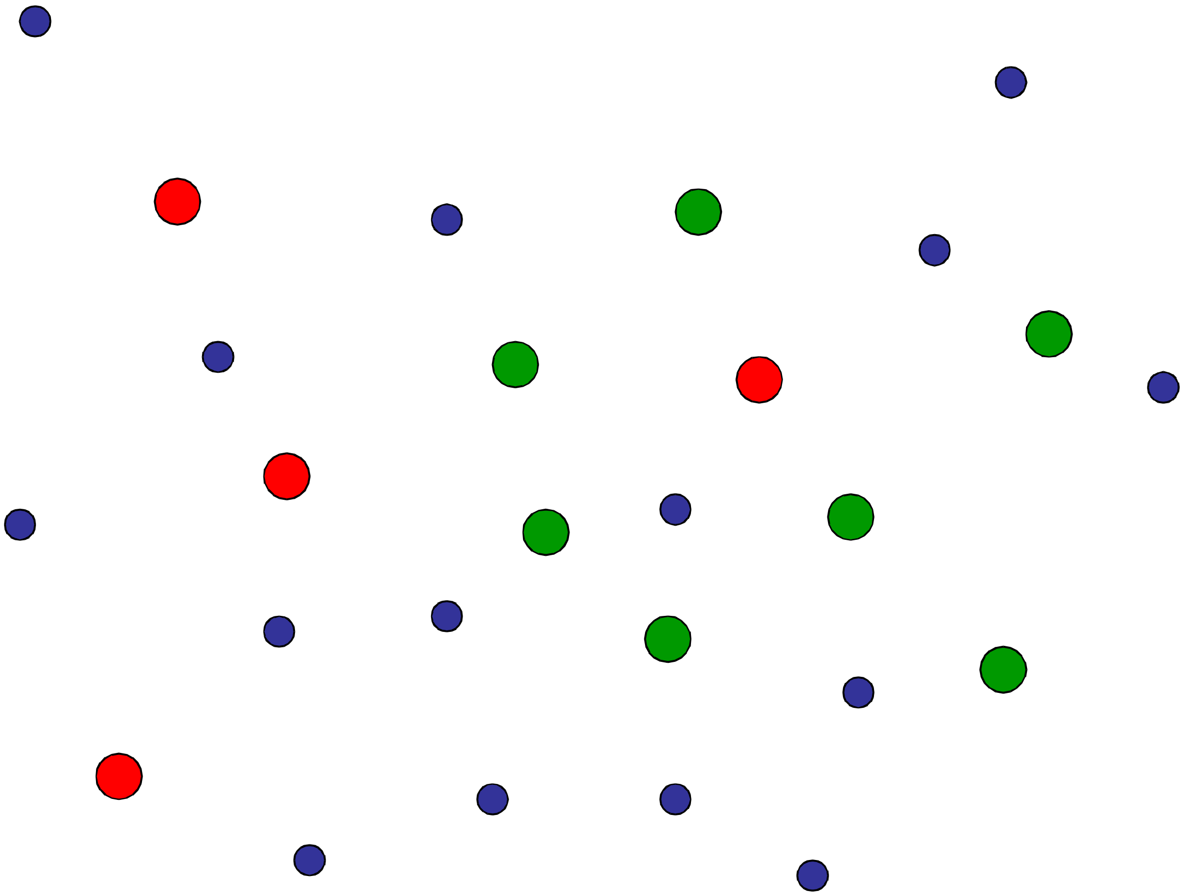
p-median problem



n (=11) potential facility locations

m (=15) users

p-median problem

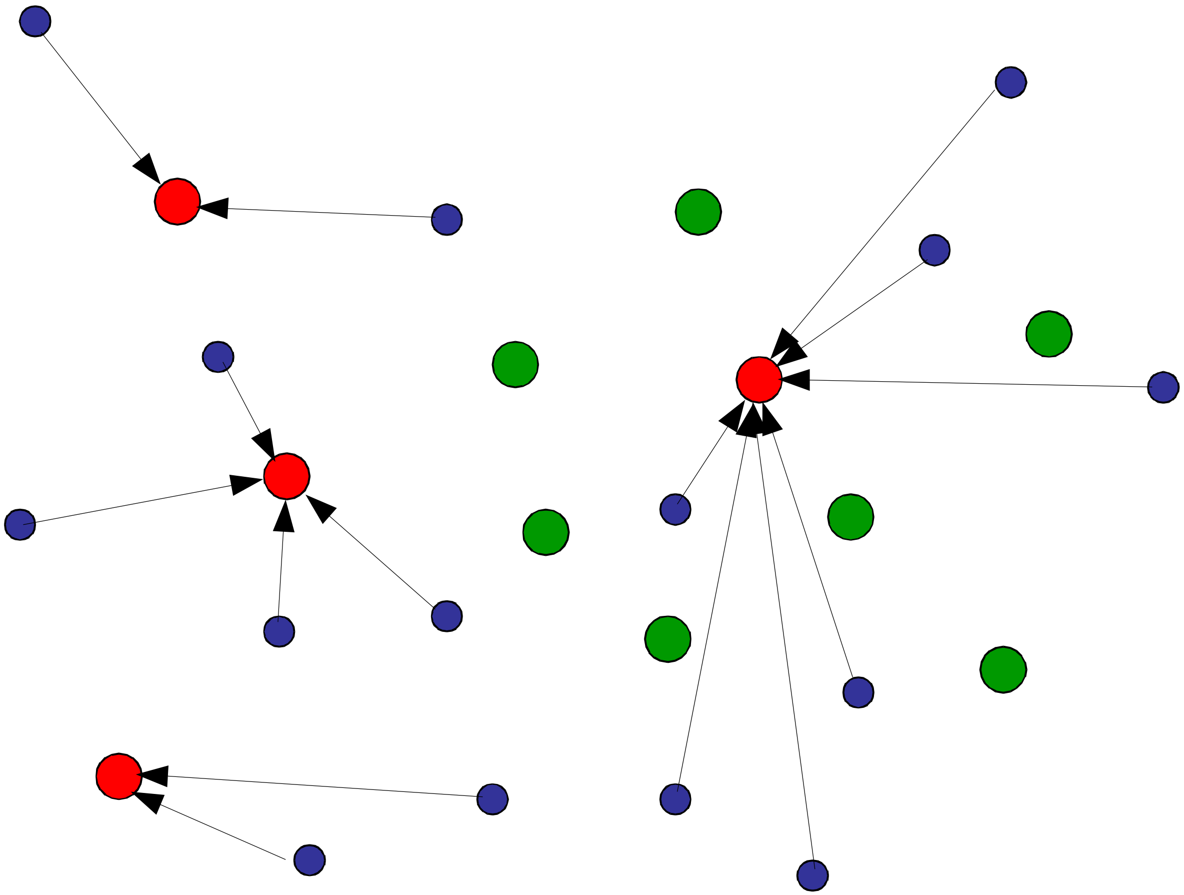


p = 4 facilities to be opened

n (=11) potential facility locations

m (=15) users

p-median problem



Users home into nearest open facility.

d(u,f) = cost of servicing user u by facility f

n (=11) potential service locations

m (=15) customers

Footprint Optimization

- In our case:
  - each exchange is a p-median user:
    - 69,534 in total (all currently covered).
  - each coordinate is a p-median facility:
    - 1035 in total (all currently open).
  - Distances: network cost.
    - (PoP rate) · (hours used by exchange)
- With p=1035, we get the current network cost.
- We want the smallest p that preserves that cost.
  - Solve the p-median problem for various values of p to find best.
  - 700 was the value we found.

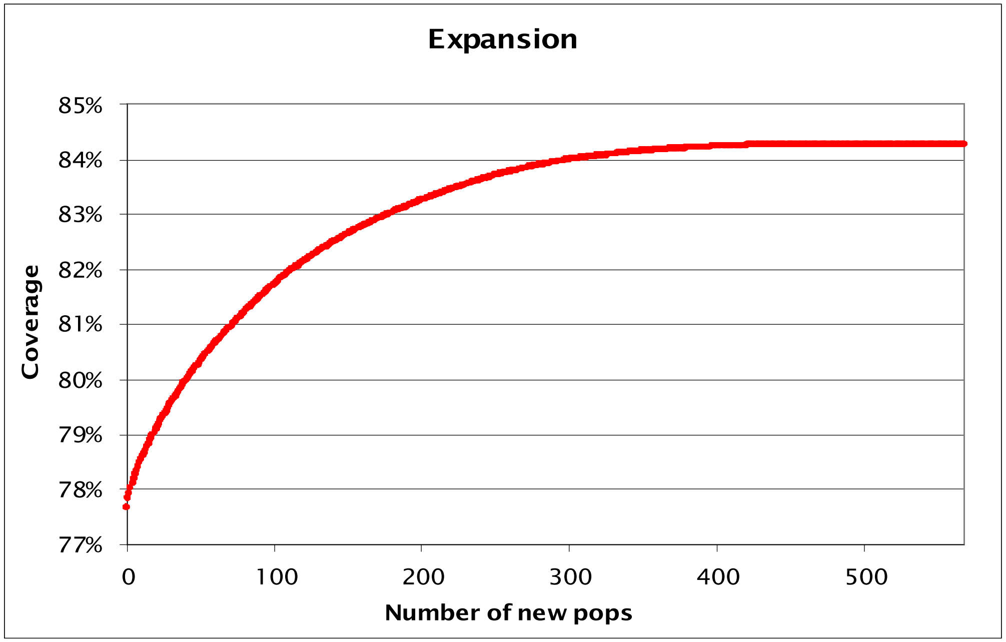
Expanding the Footprint

- Second problem:
  - Increase coverage beyond 77.66%.
- AT&T can use UUNet PoPs:
  - 1,498 candidate PoPs.
  - 568 of those cover at least one new exchange.
- Main question:
  - If we want to open p new PoPs, which PoPs do we open?
    - Goal: maximize coverage.
- This is the maximum cover problem:
  - It can be solved as a p-median problem.

From maximum cover to p-median

- Idea: minimize number of customers not covered.
  - Users:
    - exchanges not currently covered.
  - Facilities:
    - all candidate UUNet PoPs;
    - dummy facility f_0.
  - Distances:
    - $d(u,f_i) = 0$, if PoP i covers exchange u.
      - if u is covered, does not contribute to solution.
    - $d(u,f_0) =$ (# of customers in exchange u);
    - $d(u,f_i) = \infty$, if PoP i does not cover u.
      - u not covered: assigned to f_0, contributes to solution.
  - A dummy user can be used to ensure that f_0 will always belong to the solution.

Expansion



Coverage	Footprint
77.66%	current
78%	current+3
79%	current+19
80%	current+41
81%	current+72
82%	current+113
83%	current+177
84%	current+301
84.27%	current+464

Papers

- M.G.C. Resende, “Computing approximate solutions of the maximum covering problem using GRASP,” J. of Heuristics, vol. 4, pp. 161-171, 1998.
- M.G.C. Resende & R.F. Werneck, “A hybrid heuristic for the p-median problem,” J. of Heuristics, vol. 10, pp. 59-88, 2004.
- M.G.C. Resende & R.F. Werneck, “A fast swap-based local search procedure for location problems,” to appear in Annals of Operations Research, 2005.

Application 3:

Local access network design

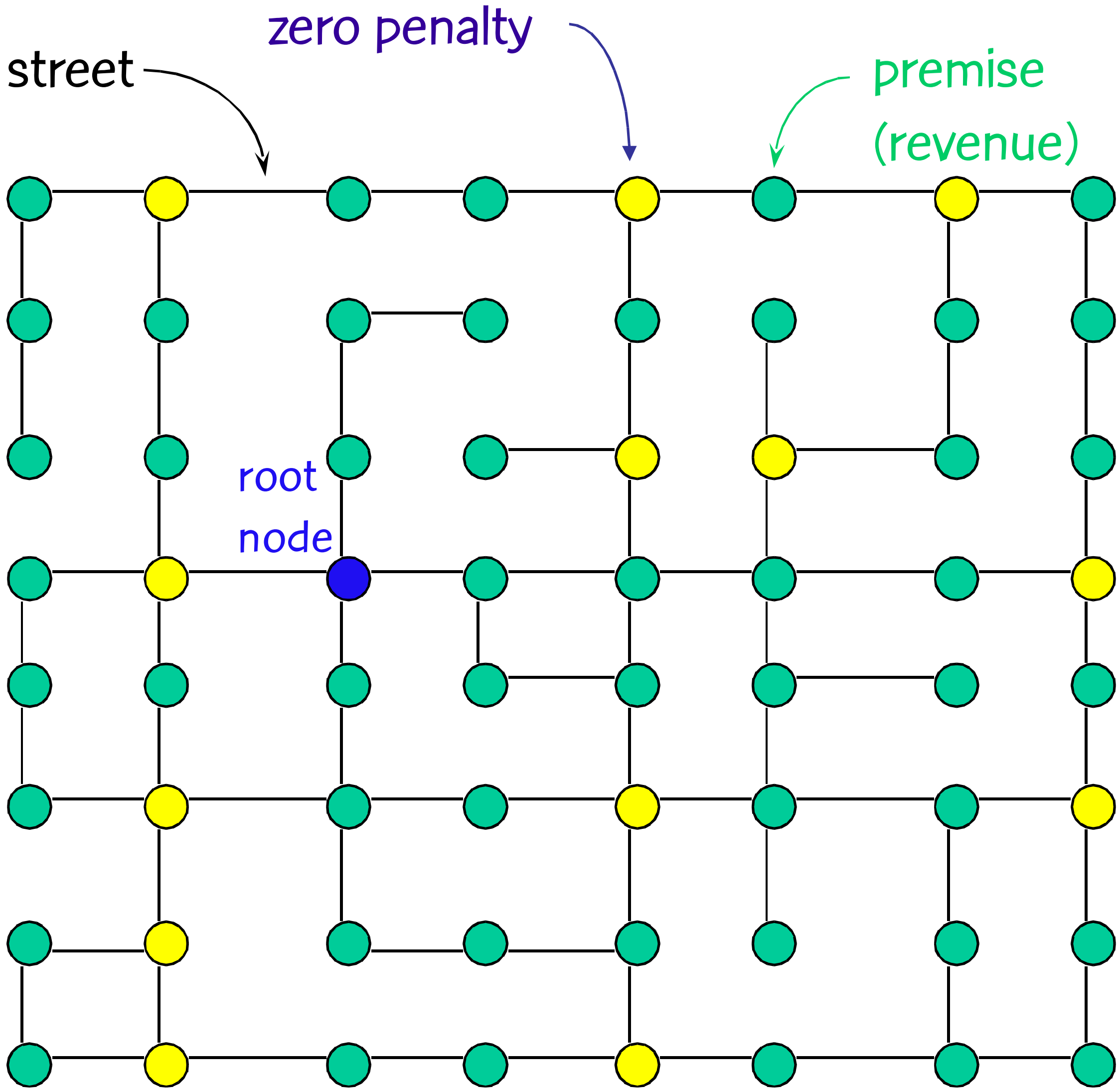
Local access network design

- Build a fiber-optic network for providing broadband connections to business and residential customers.
- Design a local access network taking into account tradeoff between:
  - cost of network
  - revenue potential of network

Local access network design

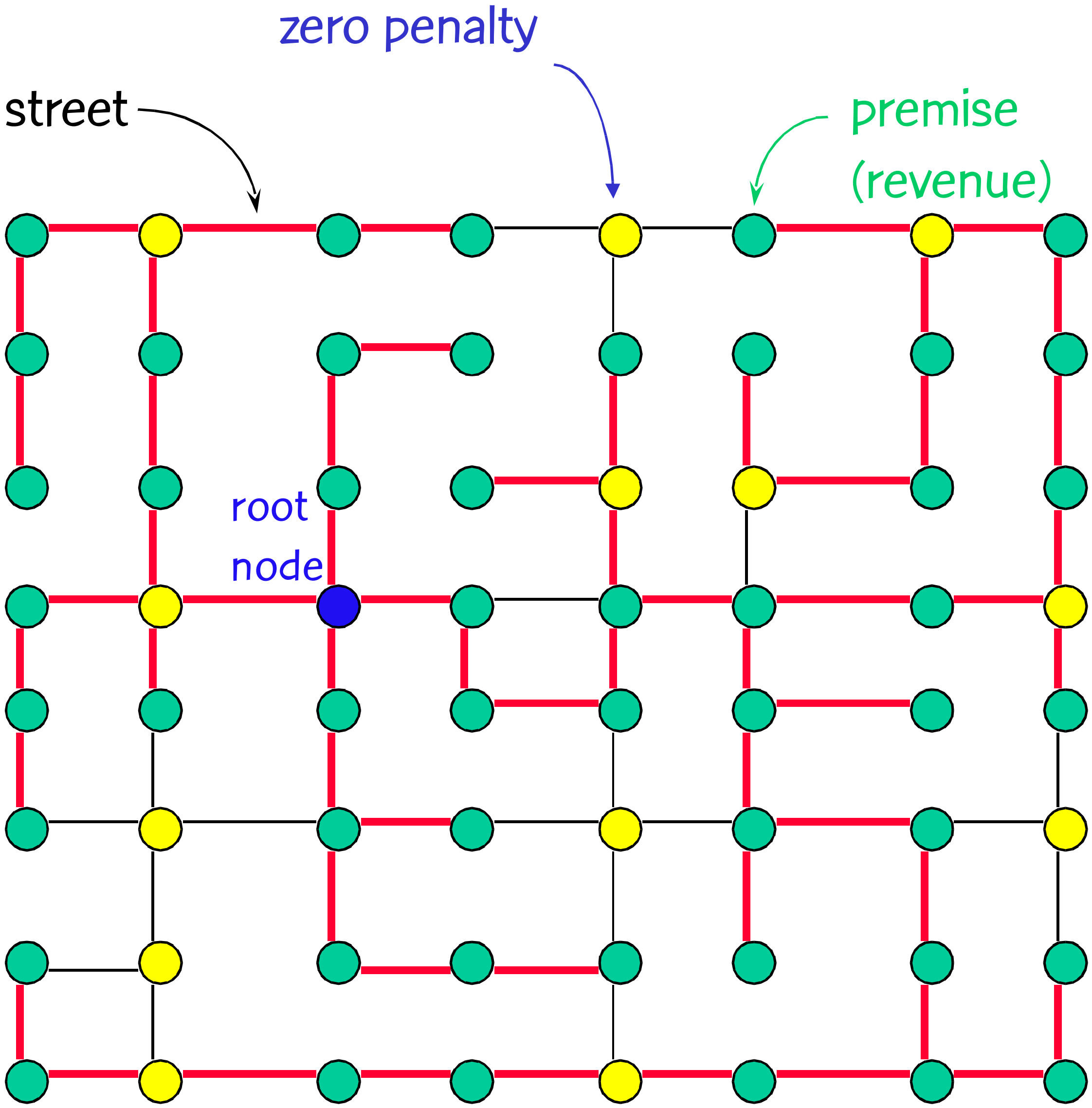
- Graph corresponds to local street map
  - Edges: street segments
    - Edge cost: cost of laying the fiber on the corresponding street segment
  - Vertices: street intersections and potential customer premises
    - Vertex penalty: estimate of potential loss of revenue if the customer were not to be serviced (intersection nodes have no penalty)

Local access network design



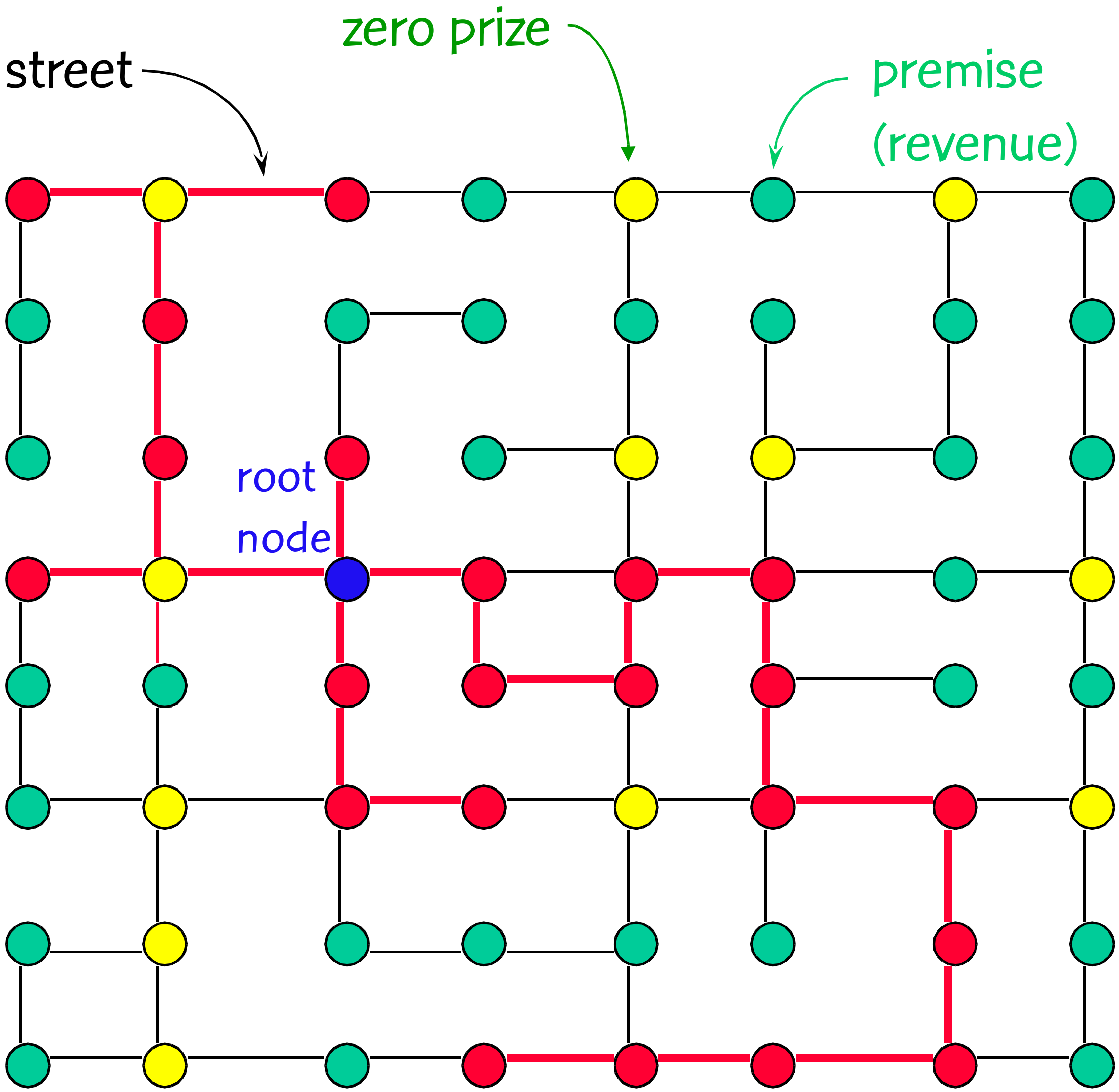
Collect all prizes

(Steiner problem in graphs)



Collect some prizes

(Prize collecting Steiner Problem in Graphs)



Multi-start heuristic

S. Canuto, M.G.C. Resende, & C.C. Ribeiro, "Local search with perturbations for the prize-collecting Steiner tree problem in graphs," Networks, vol. 38, pp. 50-58, 2001

- Repeat:
  - Perturb problem data and solve using approximation algorithm of Goemans and Williamson (1996);
  - If solution is new, perform swap-based local search;
  - Attempt to insert solution into POOL;
  - Select solution at random from POOL and explore path from current iterate and POOL solution using path-relinking;
- Starting from best POOL solution, apply variable neighborhood search;

A cutting planes algorithm: Lower bound

A. Lucena & M.G.C. Resende, “Strong lower bounds for the prize collecting Steiner tree problem in graphs,” Discrete Applied Mathematics, vol. 141, pp. 277-294, 2004.

- Integer programming (IP) formulation
- Cutting planes algorithm to solve linear programming relaxation of IP

Computational results

- 114 test problems
  - Smallest instance: 100 nodes & 284 edges
  - Largest instance: 1000 nodes & 25,000 edges
  - Three classes:
    - Johnson, Minkoff, & Phillips (1999) P & K problems
    - Steiner C problems (derived from SPG Steiner C test problems in OR-Library)
    - Steiner D problems (derived from SPG Steiner D test problems in OR-Library)

Computational results:

Cutting planes algorithm

- Found optimal LP solutions in 97 of the 114 test problems (85%)
- Found tight lower bounds (equal to best known upper bounds) in 104 instances (91%)
- Of the 97 optimal LP solutions, 94 were integral. Each of the 3 fractional solutions was off of the best known upper bound by less than ½
- On the 12 instances for which tight lower bounds were not produced, the bounds produced had at most a 1.3% deviation from the best known upper bounds
- In 13 of the 114 instances, single vertex optima were found
- In 7 instances the algorithm took over 100,000 seconds to converge to a lower bound. The longest run took over 10 CPU days.

Computational results:

heuristic upper bounds

- Heuristic found
  - 89 of 104 known optimal values (86%)
  - solution within 1% of lower bound for 104 of 114 problems

Number of optima found with each additional heuristic

Type	num	GW	+LS	+PR+VNS		tot
C	38	6	2	25	3	36
D	32	5	6	10	4	25
JMP	34	8	6	12	2	28

104 89

Computational results:

heuristic upper bounds

Number of instances with given relative error

heuristic	< 1%	< 5%	< 10%	max (%)
GW	7	22	29	36.4
+LS	17	34	37	11.1
+PR	35	38	40	9.1
+VNS	38	40	40	1.1

Problem type Steiner C

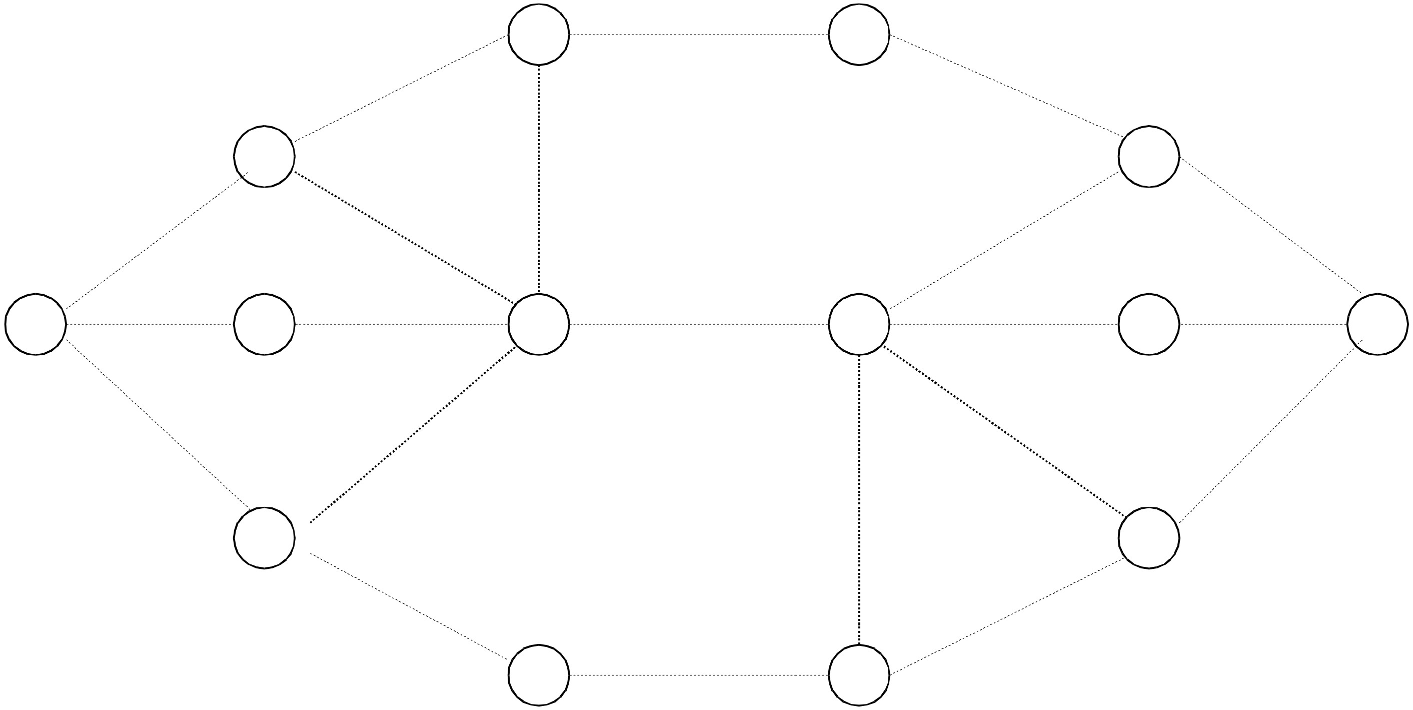
Application 4:

Traffic routing on a virtual private network

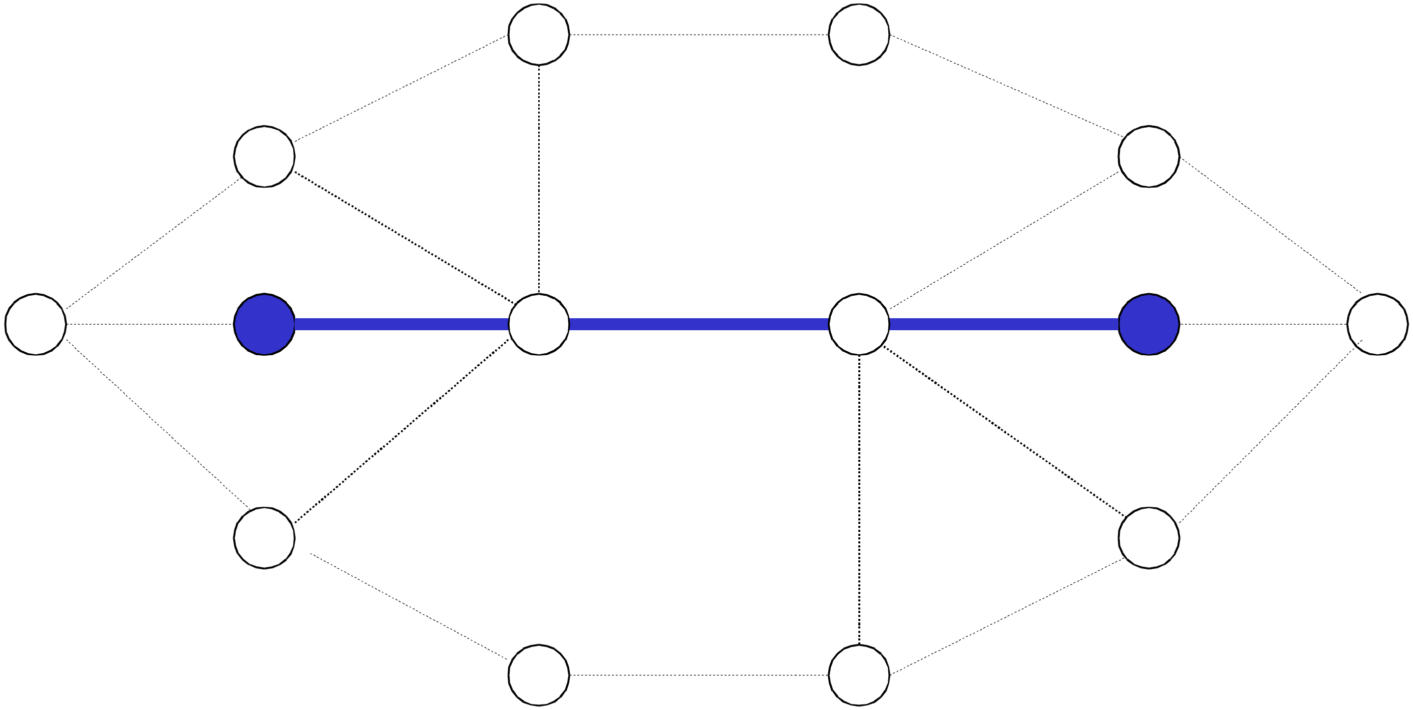
Traffic routing on a virtual private network

- Frame relay service offers virtual private networks to customers by providing long-term private virtual circuits (PVCs) between customer endpoints on a backbone network.
- Routing is done either automatically by switch or by the network designer without any knowledge of future requests.
- Over time, these decisions cause inefficiencies in the network and occasionally offline rerouting (grooming) of the PVCs is needed:
  - integer multicommodity network flow problem: Resende & Ribeiro (2003)

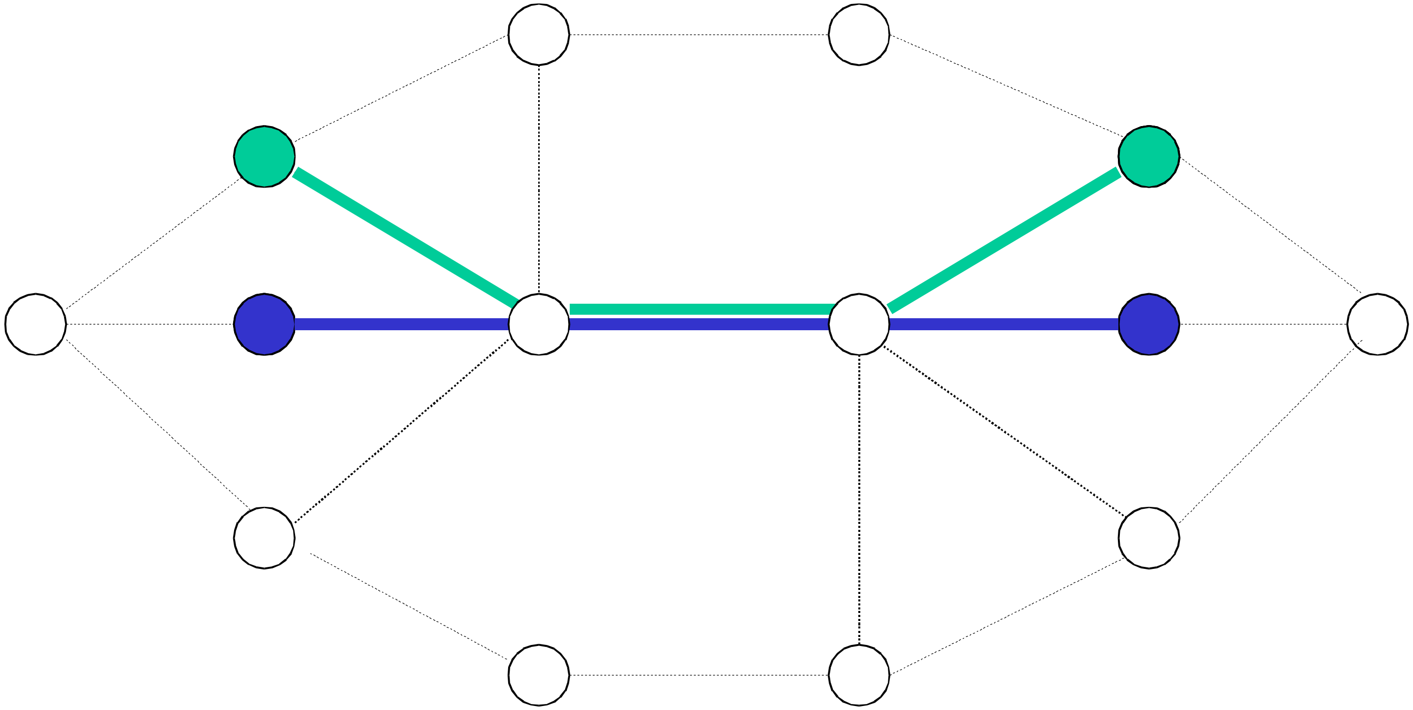
Traffic routing on a virtual private network



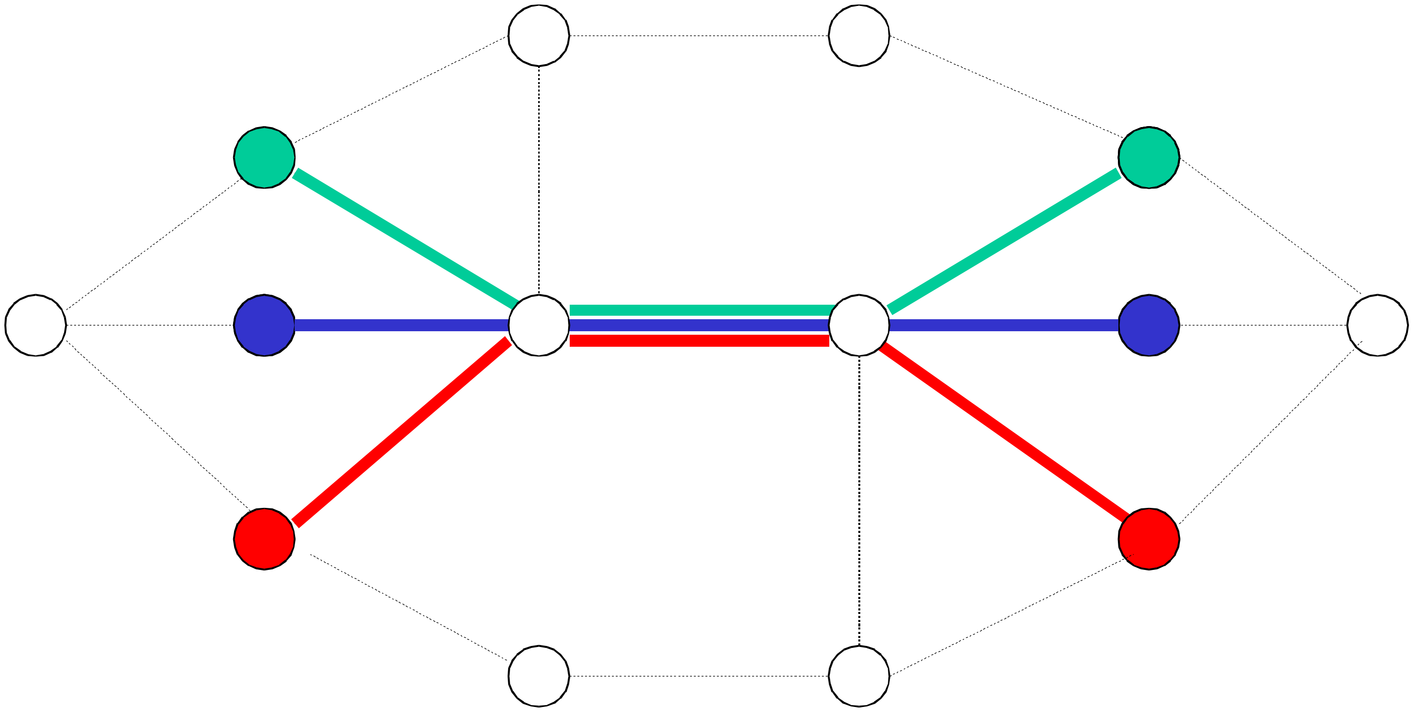
Traffic routing on a virtual private network



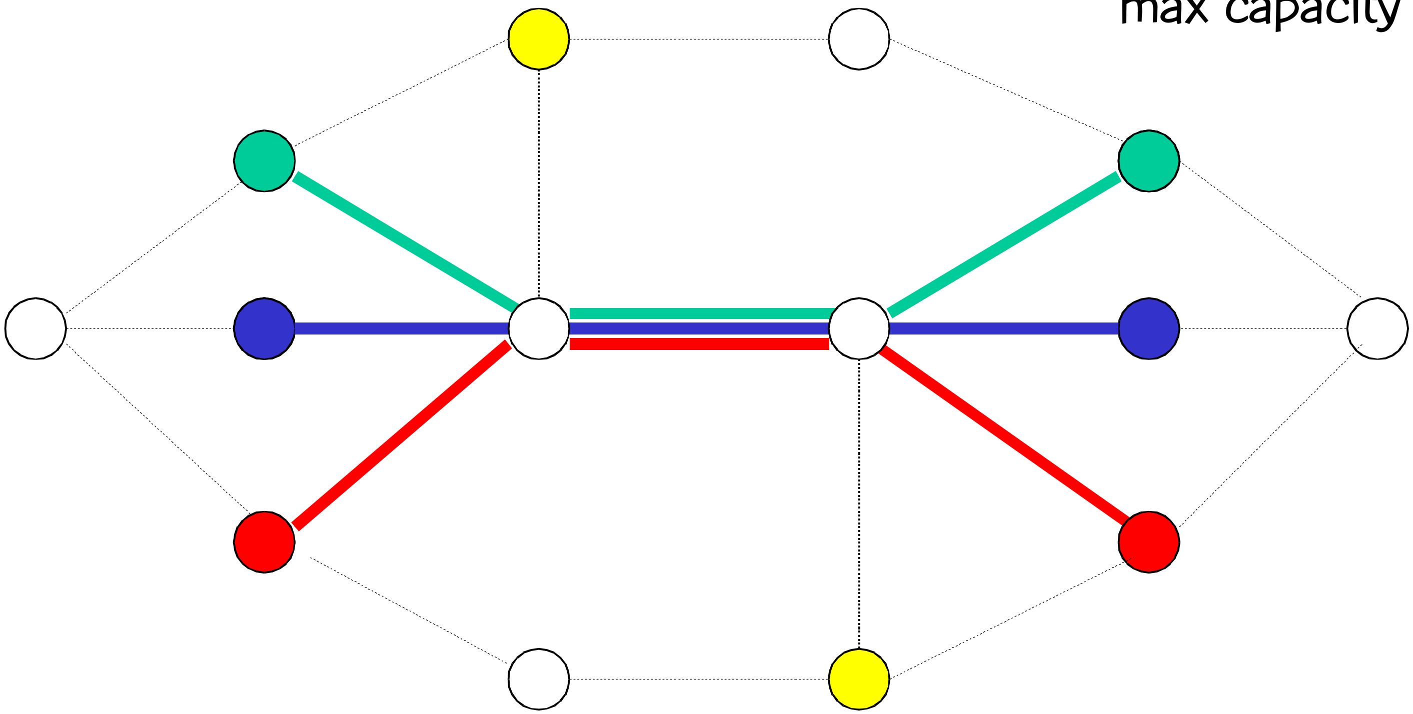
Traffic routing on a virtual private network



Traffic routing on a virtual private network

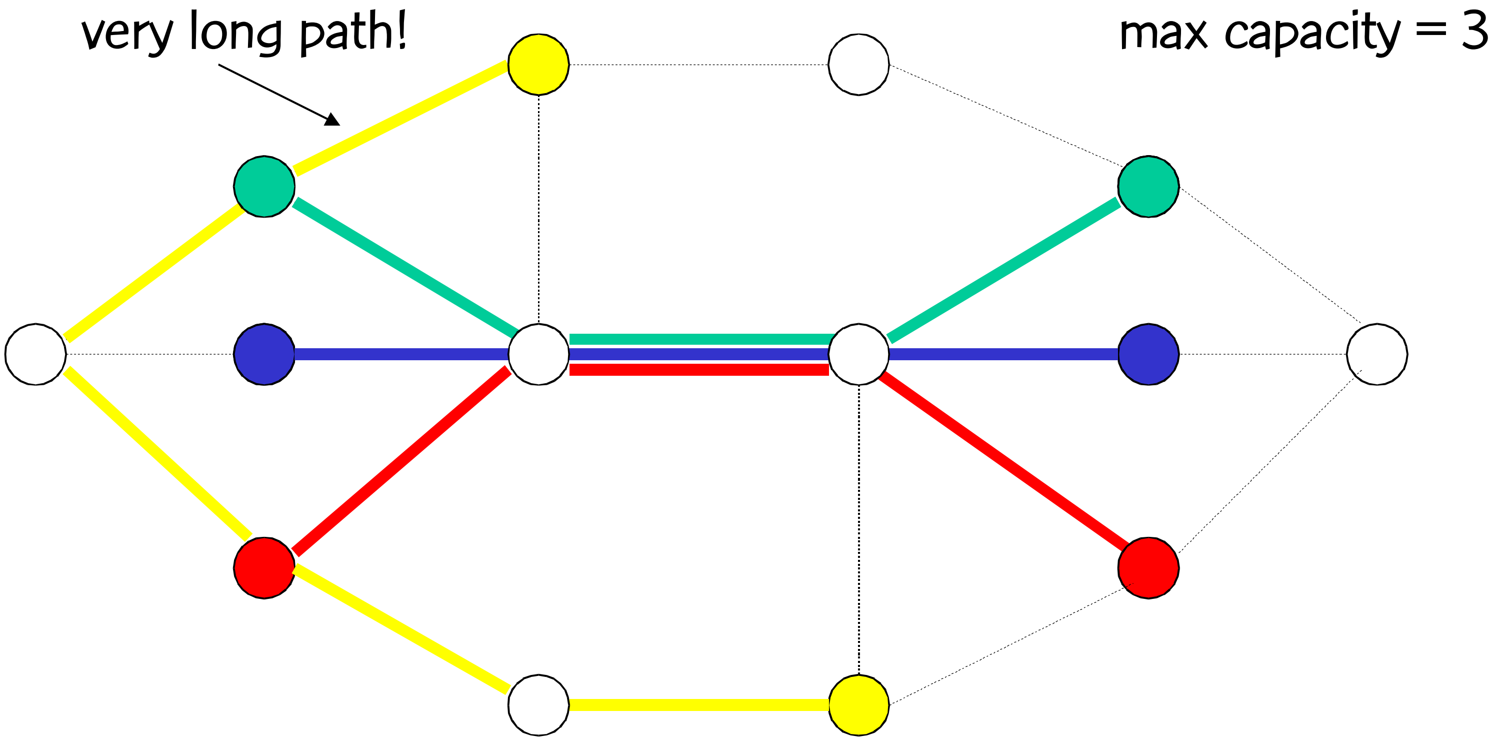


Traffic routing on a virtual private network

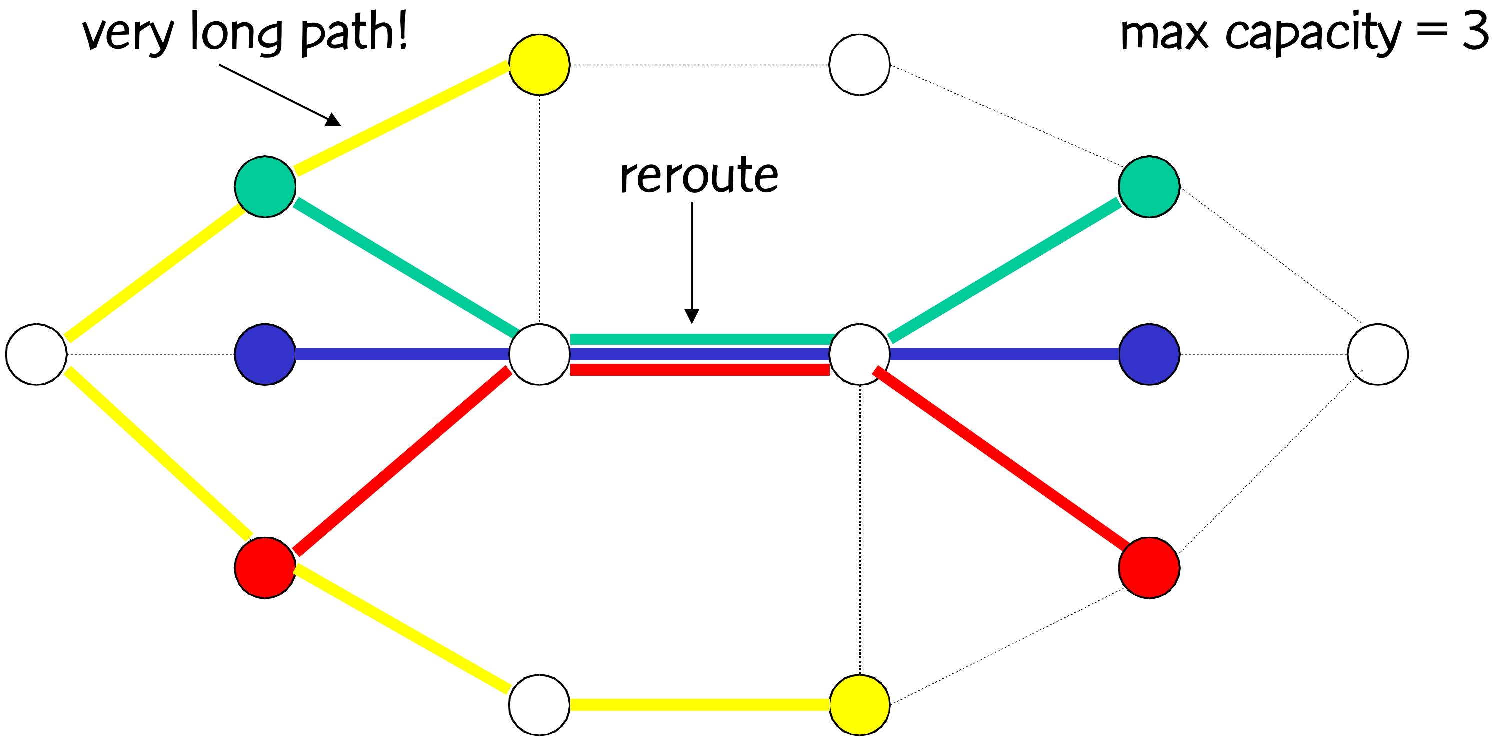


max capacity = 3

Traffic routing on a virtual private network



Traffic routing on a virtual private network

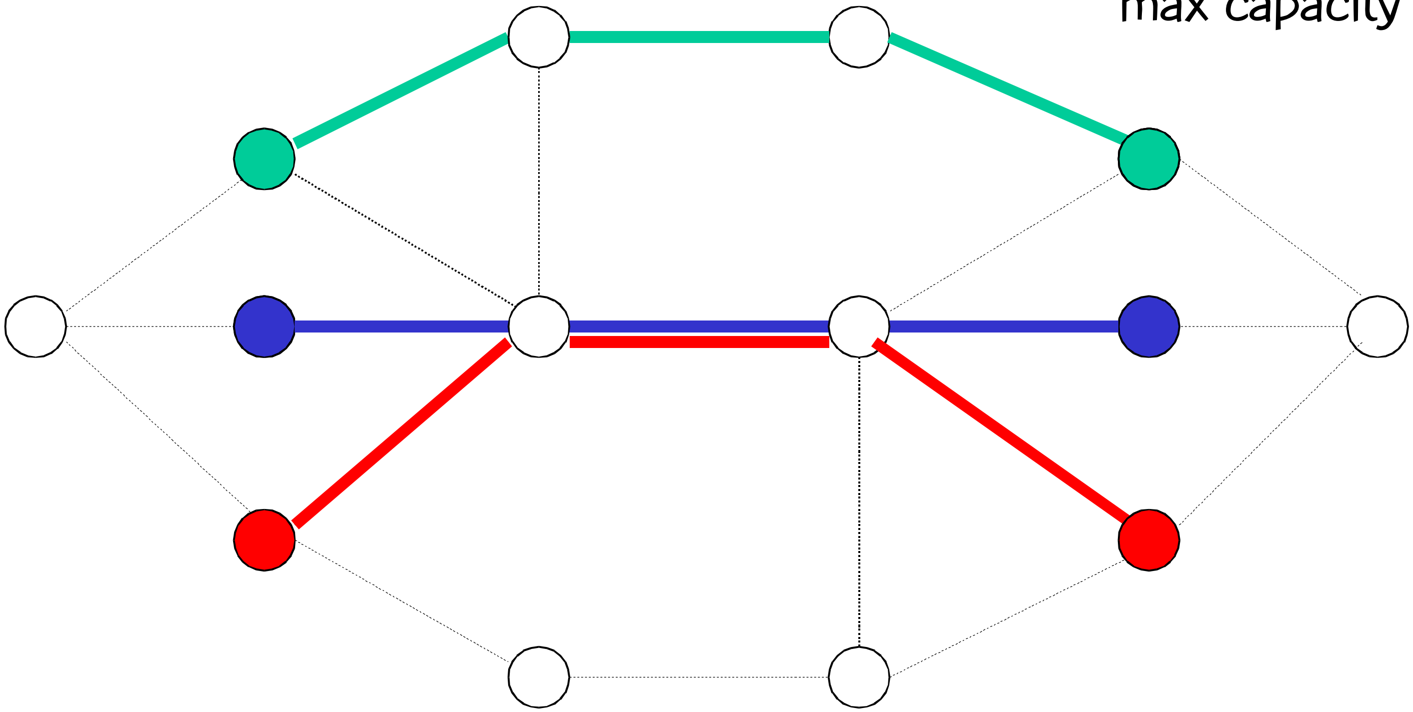


very long path!

reroute

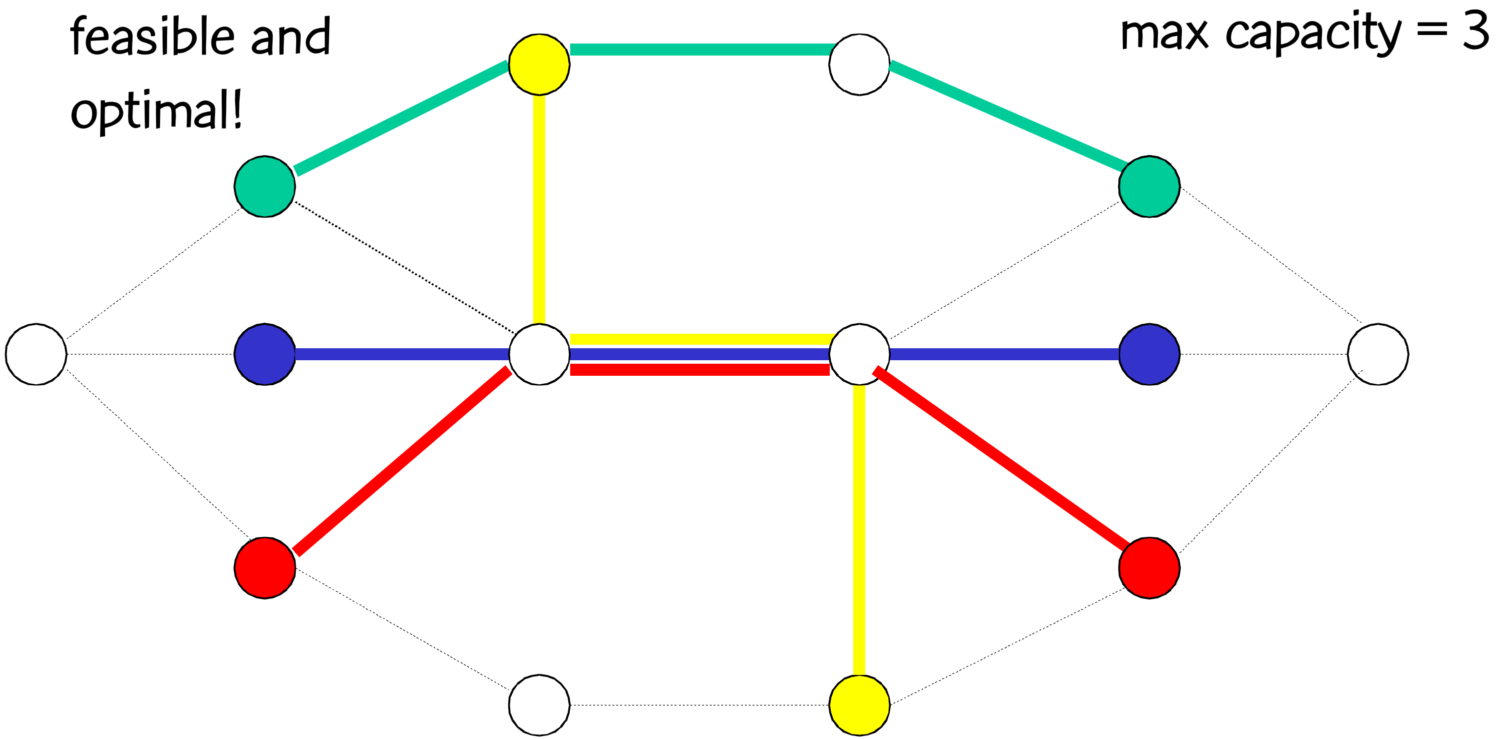
max capacity = 3

Traffic routing on a virtual private network



max capacity = 3

Traffic routing on a virtual private network



feasible and optimal!

max capacity = 3

Papers

- M.G.C. Resende & C.C. Ribeiro, “A GRASP with path-relinking for private virtual circuit routing,” Networks, vol. 41, pp. 104-114, 2003.
- M.G.C. Resende & C.C. Ribeiro, “GRASP with path-relinking: Recent advances and applications,” in “Metaheuristics: Progress as Real Problem Solvers,” Ibaraki, Nonobe and Yagiura, (Eds.), Springer, 2005.

Application 5: Internet traffic engineering

Internet traffic engineering

- Internet traffic has been doubling each year [Coffman & Odlyzko, 2001]
- In the1995-96 period, there was a doubling of traffic each three months!
  - Web browsers were introduced.
- Increasingly heavy traffic (due to video, voice, etc.) will raise the requirements of the Internet of tomorrow.

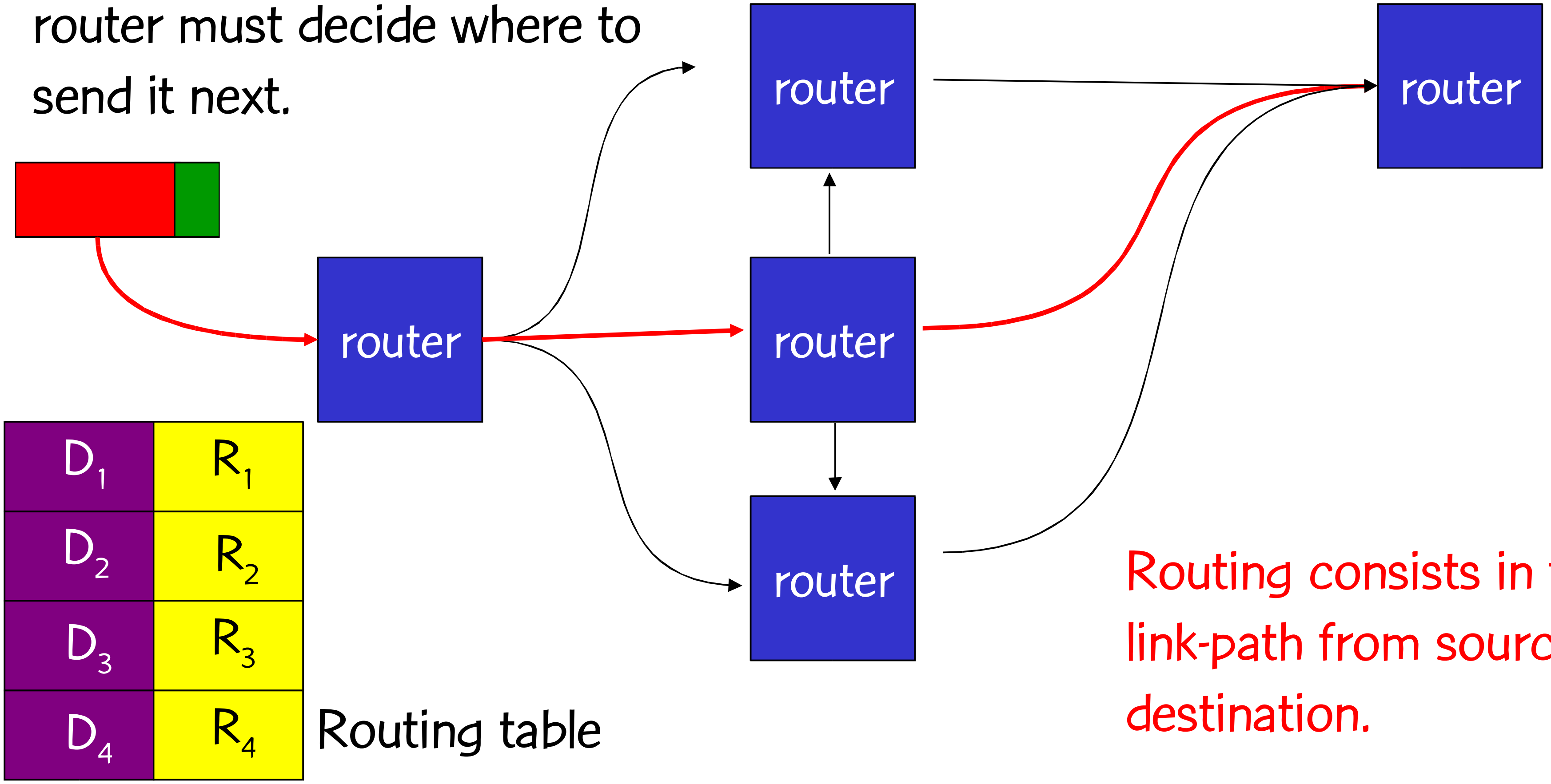
Internet traffic engineering

- Objective: make more efficient use of existing network resources.
- Routing of traffic can have a major impact on efficiency of network resource utilization.

Packet routing

When packet arrives at router, router must decide where to send it next.

Packet's final destination.



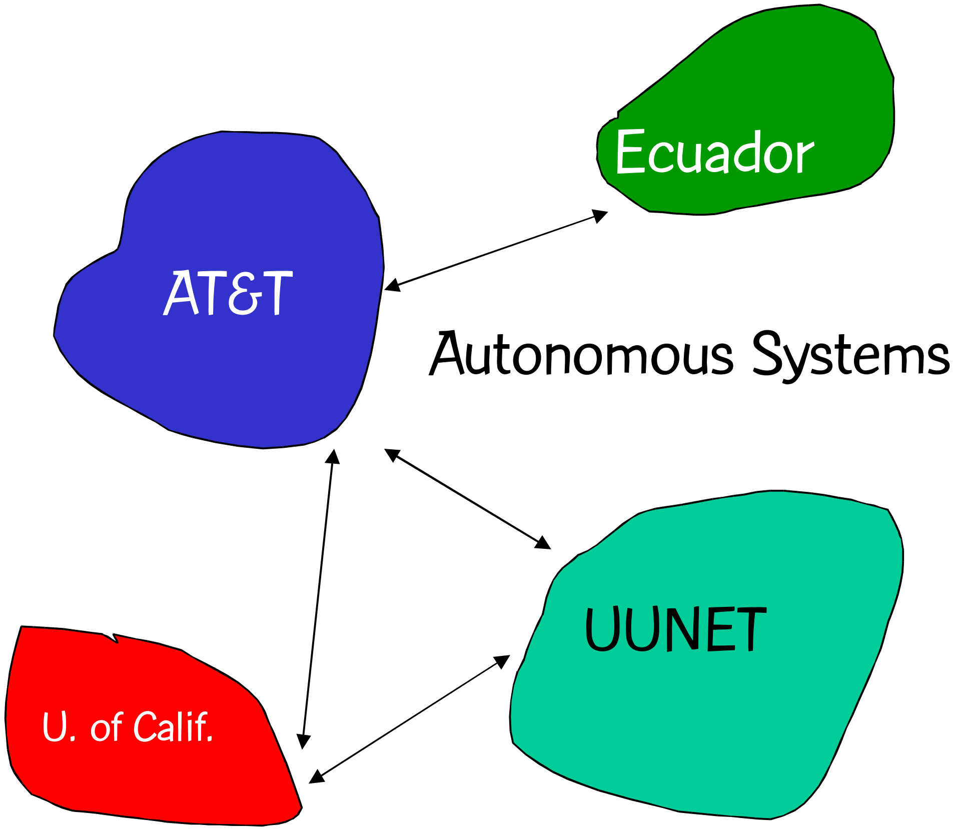
D_1	R_1
D_2	R_2
D_3	R_3
D_4	R_4

Routing table

Routing consists in finding a link-path from source to destination.

OSPF (Open Shortest Path First)

- OSPF is a commonly used intra-domain routing protocol (IGP).
- Routers exchange routing information with all other routers in the autonomous system (AS).
  - Complete network topology knowledge is available to all routers, i.e. state of all routers and links in the AS.



OSPF routing

- Assign an integer weight | [1, w_{max}] to each link in AS.

  In general, $w_{max} = 65535 = 2^{16} - 1$.

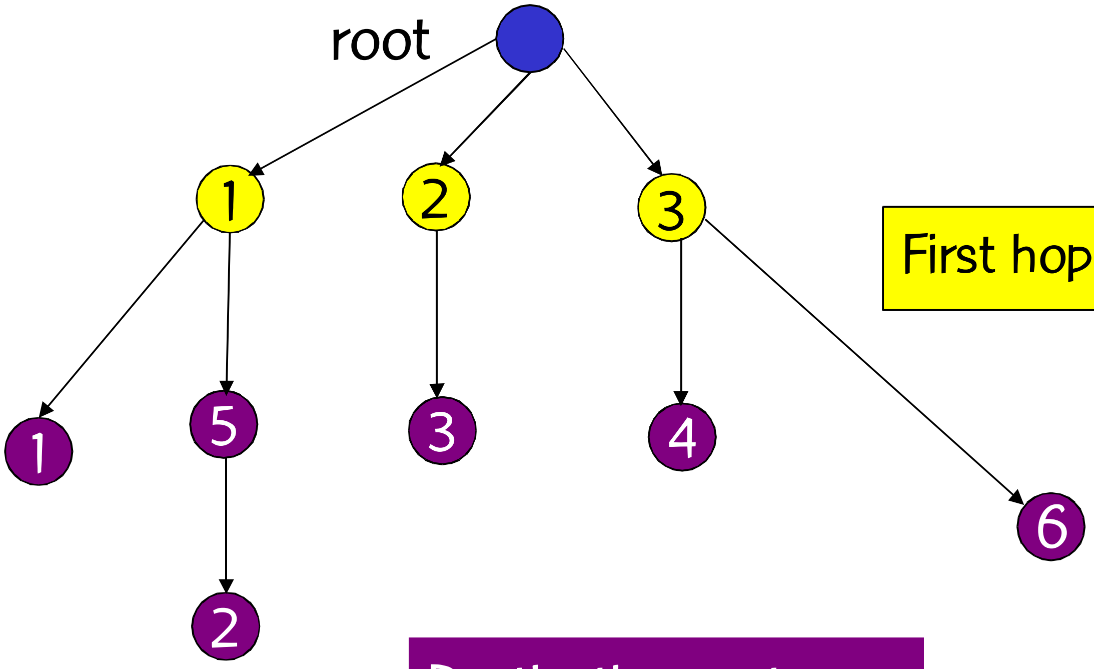
- Each router computes tree of shortest weight paths to all other routers in the AS, with itself as the root, using Dijkstra's algorithm.

OSPF routing

Routing table

D_1	R_1
D_2	R_1
D_3	R_2
D_4	R_3
D_5	R_1
D_6	R_3

Routing table is filled
with first hop routers
for each possible destination.



First hop routers.

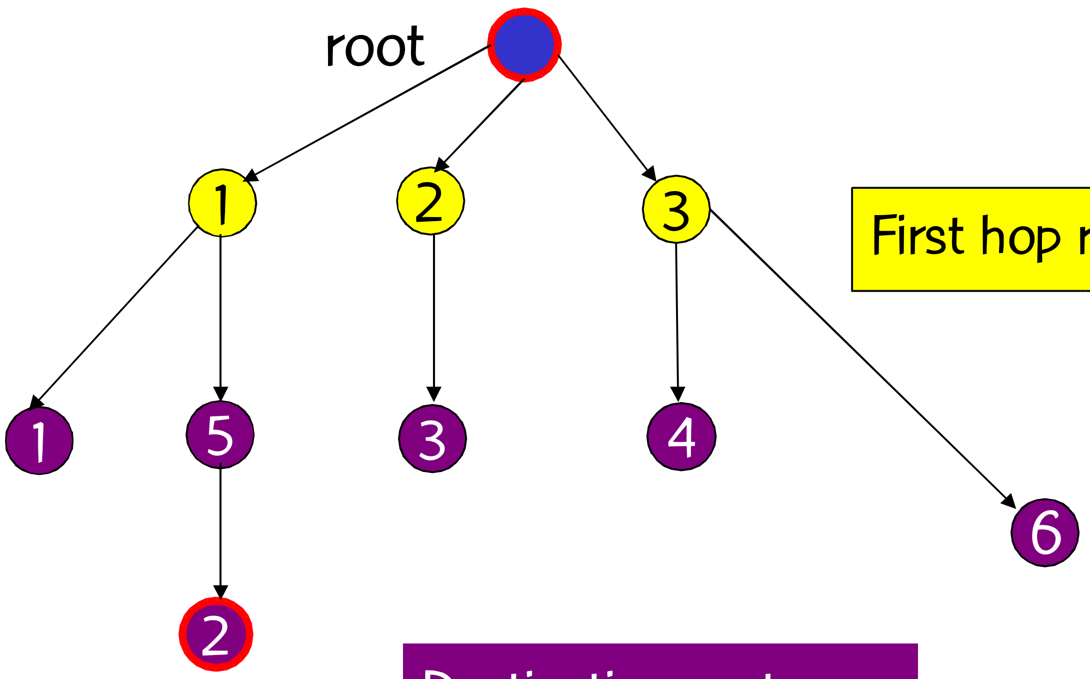
Destination routers

OSPF routing

Routing table is filled
with first hop routers
for each possible destination.

Routing table

D_1	R_1
D_2	R_1
D_3	R_2
D_4	R_3
D_5	R_1
D_6	R_3



First hop routers.

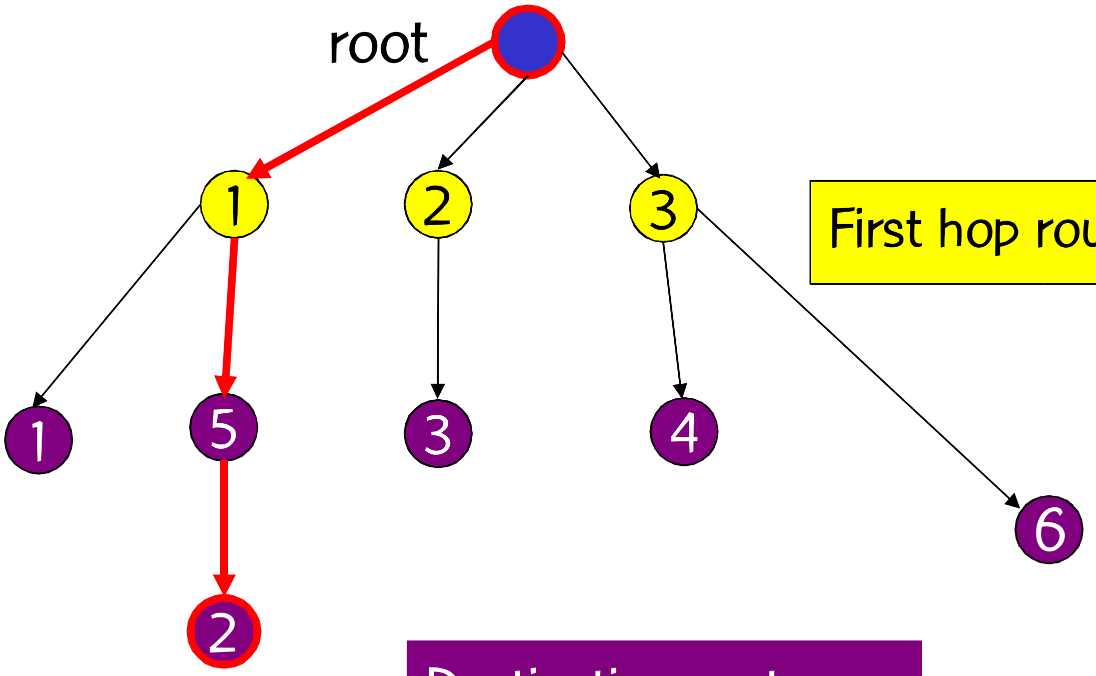
Destination routers

OSPF routing

Routing table is filled
with first hop routers
for each possible destination.

Routing table

D_1	R_1
D_2	R_1
D_3	R_2
D_4	R_3
D_5	R_1
D_6	R_3



First hop routers.

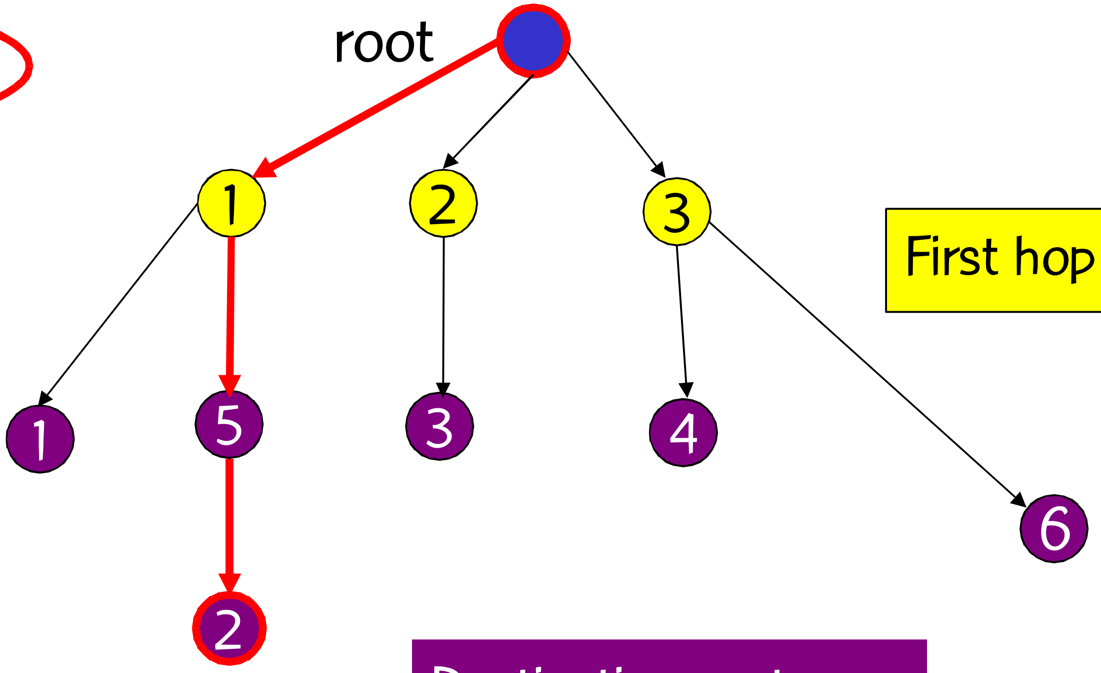
Destination routers

OSPF routing

Routing table is filled
with first hop routers
for each possible destination.

Routing table

D_1	R_1
D_2	R_1
D_3	R_2
D_4	R_3
D_5	R_1
D_6	R_3



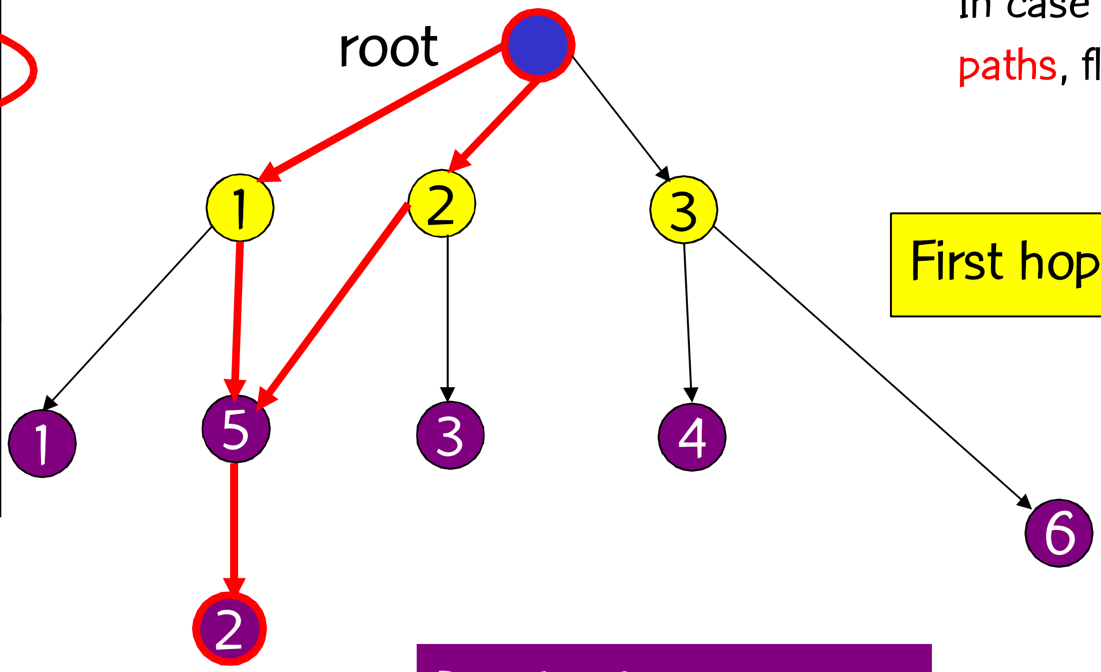
First hop routers.

Destination routers

OSPF routing

Routing table

D_1	R_1
D_2	$R_{1,}$ R_2
D_3	R_2
D_4	R_3
D_5	R_1
D_6	R_3



Routing table is filled
with first hop routers
for each possible destination.
In case of multiple shortest
paths, flow is evenly split.

First hop routers.

Destination routers

OSPF weight setting

- OSPF weights are assigned by network operator.
  - CISCO assigns, by default, a weight proportional to the inverse of the link bandwidth (Inv Cap).
  - If all weights are unit, the weight of a path is the number of hops in the path.
- We propose a hybrid genetic algorithm to find good OSPF weights.
  - Memetic algorithm
  - Genetic algorithm with optimized crossover

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 $a \mid A$.
- We use the measure of Fortz & Thorup (2000) to compute congestion:

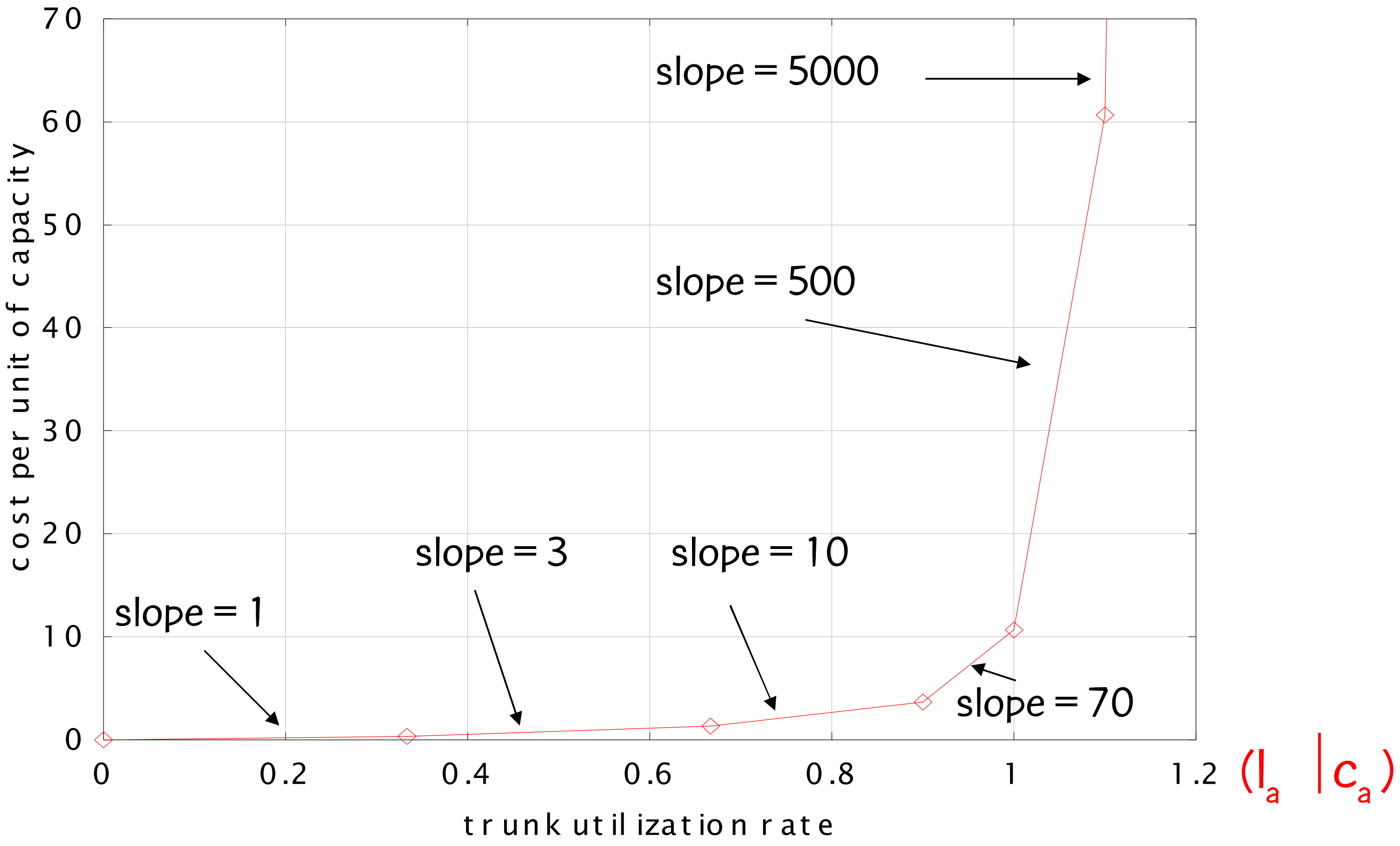
$$| = |_1(l_1) + |_2(l_2) + \ldots + |_{|A|}(l_{|A|})$$

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

$|_a(l_a)$ is piecewise linear and convex,

$|_a(0) = 0$, for all $a \mid A$.

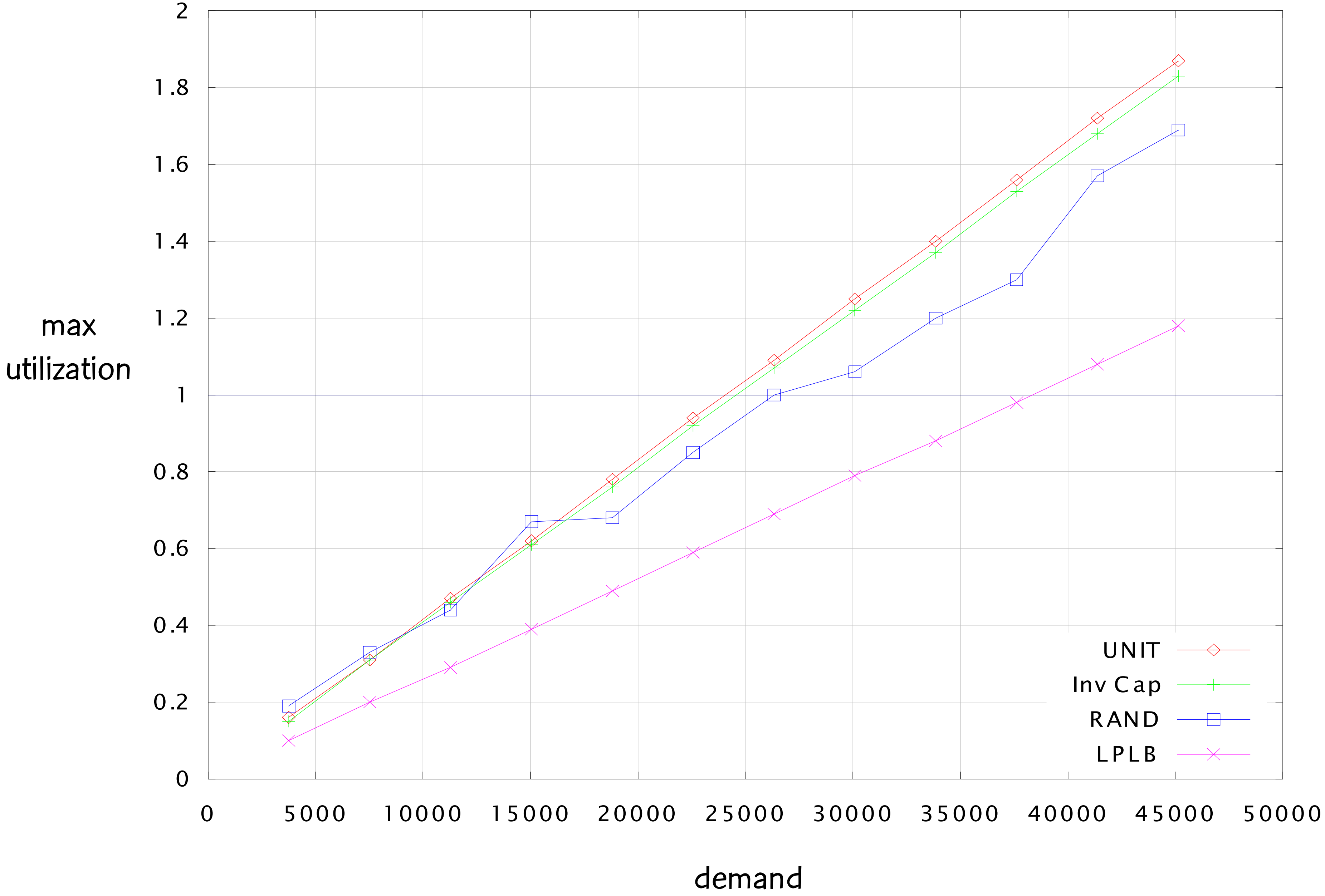
Piecewise linear and convex $|_a(l_a)$ link congestion measure



OSPF weight setting problem

- Given a directed network $G = (N, A)$ with link capacities c_a | A and demand matrix $D = (d_{s,t})$ specifying a demand to be sent from node s to node t :
  - Assign weights w_a | $[1, w_{max}]$ to each link a | A, such that the objective function | is minimized when demand is routed according to the OSPF protocol.

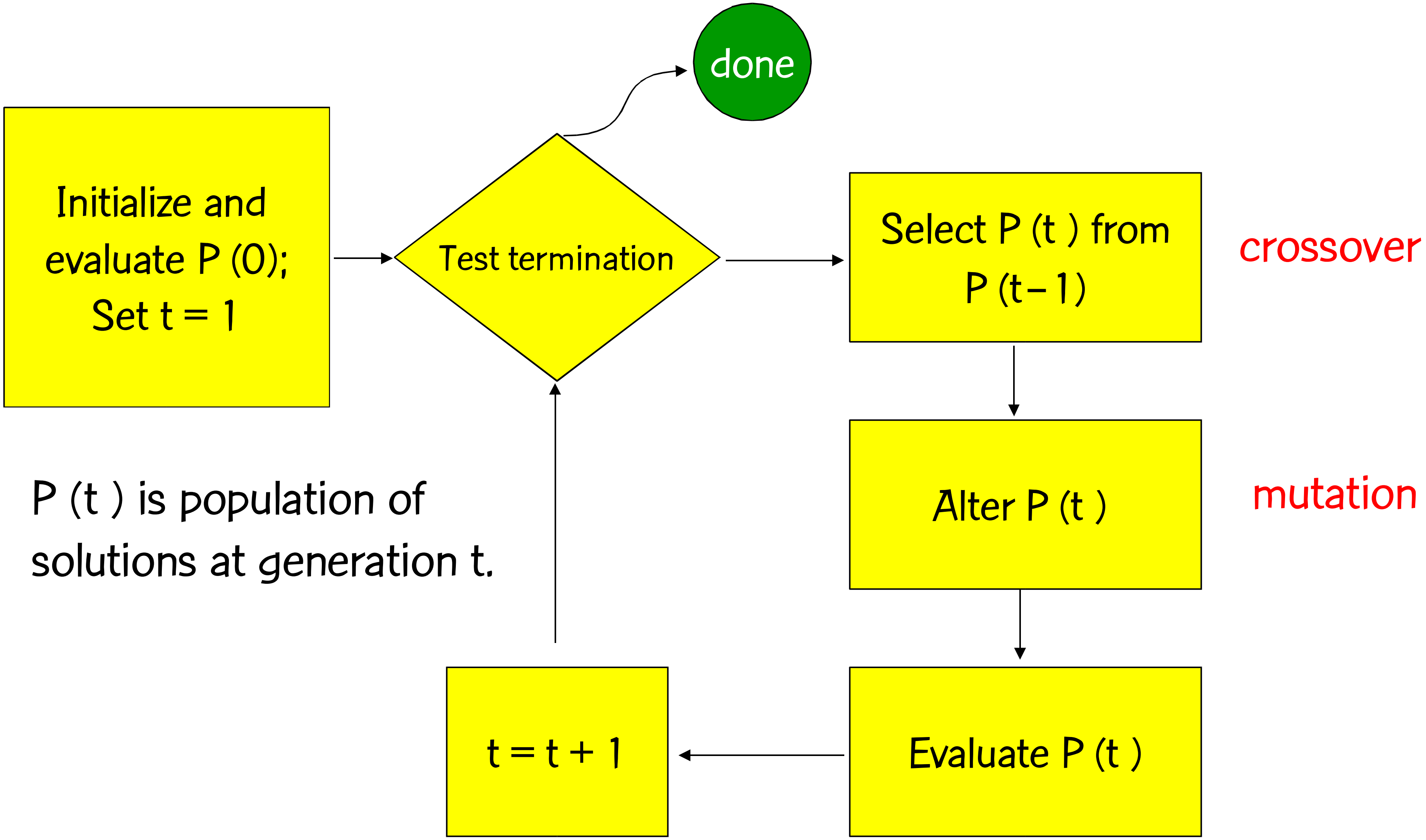
AT&T Worldnet backbone network (90 routers, 274 links)



Genetic and hybrid genetic algorithms for OSPF weight setting problem

- Genetic
  - M. Ericsson, M.G.C. Resende, & P.M. Pardalos, " A genetic algorithm for the weight setting problem in OSPF routing, J. of Combinatorial Optimization, vol. 6, pp. 299-333, 2002.
- Hybrid genetic
  - L.S. Buriol, M.G.C. Resende, C.C. Ribeiro, & M. Thorup, "A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing," to appear in Networks, 2005.

Genetic algorithms



P (t) is population of solutions at generation t.

Solution encoding

- A population consists of nPop = 50 integer weight arrays: $w = (w_1, w_2 ,..., w_{|A|})$,

  where $w_a \; | \; [1, w_{max} = 20]$

- All possible weight arrays correspond to feasible solutions.

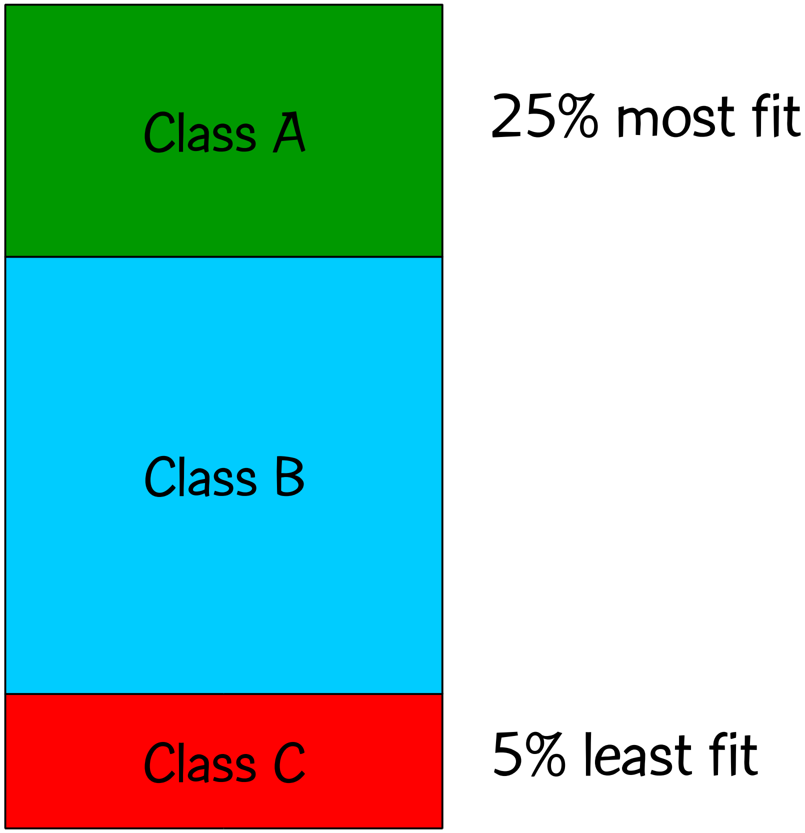
Initial population

- nPop solutions, with each weight randomly generated, uniformly in the interval $[1, w_{max}/3]$.

Solution evaluation

- For each demand pair (s,t), route using OSPF, computing demand pair loads $l_a^{s,t}$ on each link a | A.
- Add up demand pair loads on each link a | A, yielding total load l_a on link.
- Compute link congestion cost $|_a(l_a)$ for each link a | A.
- Add up costs: $| = |_1(l_1) + |_2(l_2) + \ldots + |_{|A|}(l_{|A|})$

Population partitioning



Class A

Class B

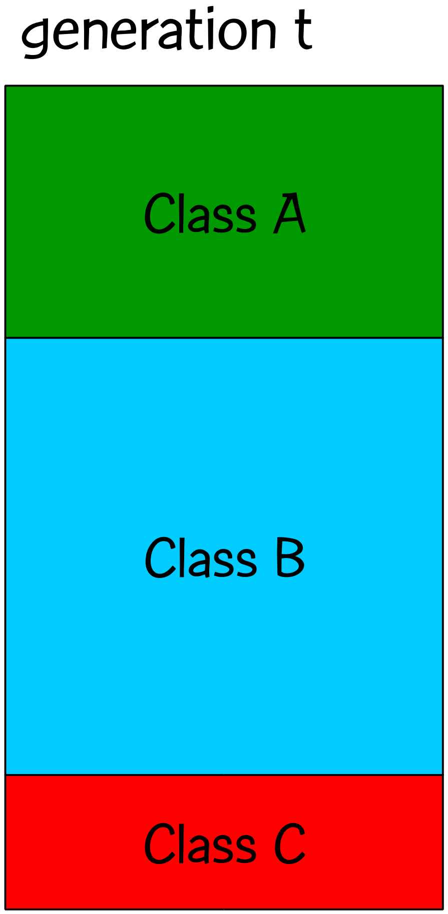
Class C

25% most fit

5% least fit

Population is sorted according to solution value and solutions are classified into three categories.

Population dynamics



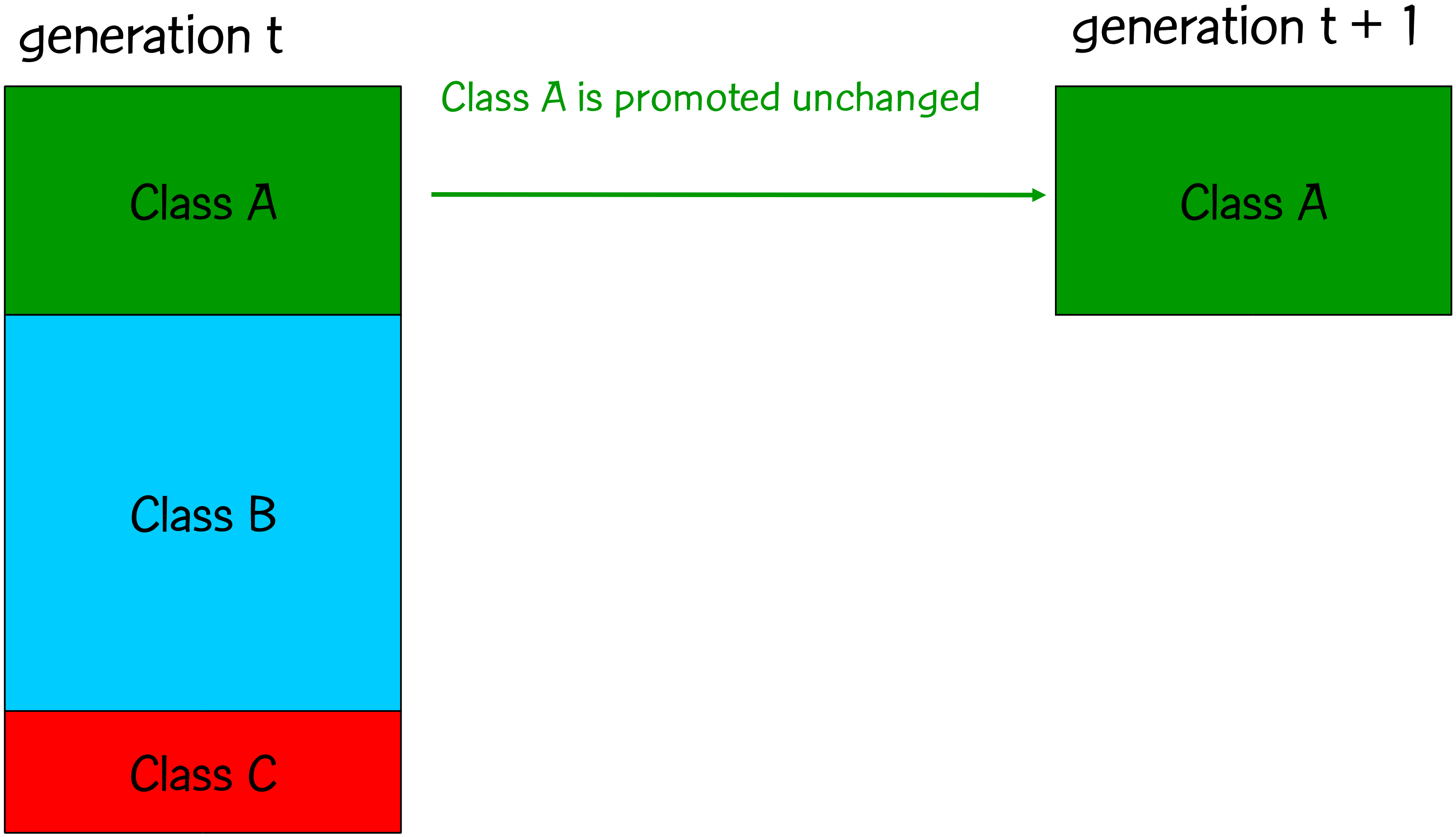
generation t

Class A

Class B

Class C

Population dynamics



generation t

generation t + 1

Class A is promoted unchanged

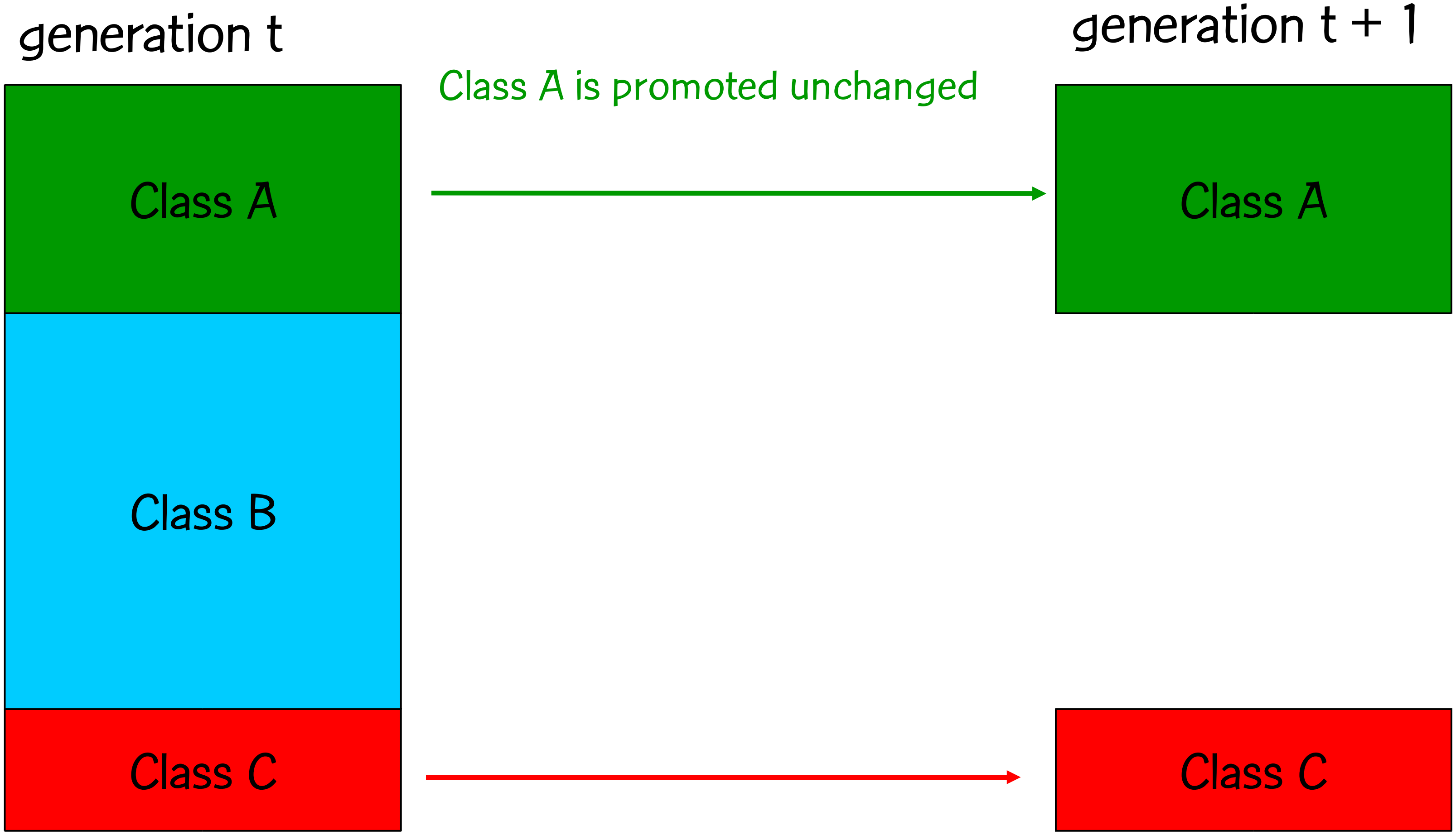
Class A

Class B

Class C

Class A

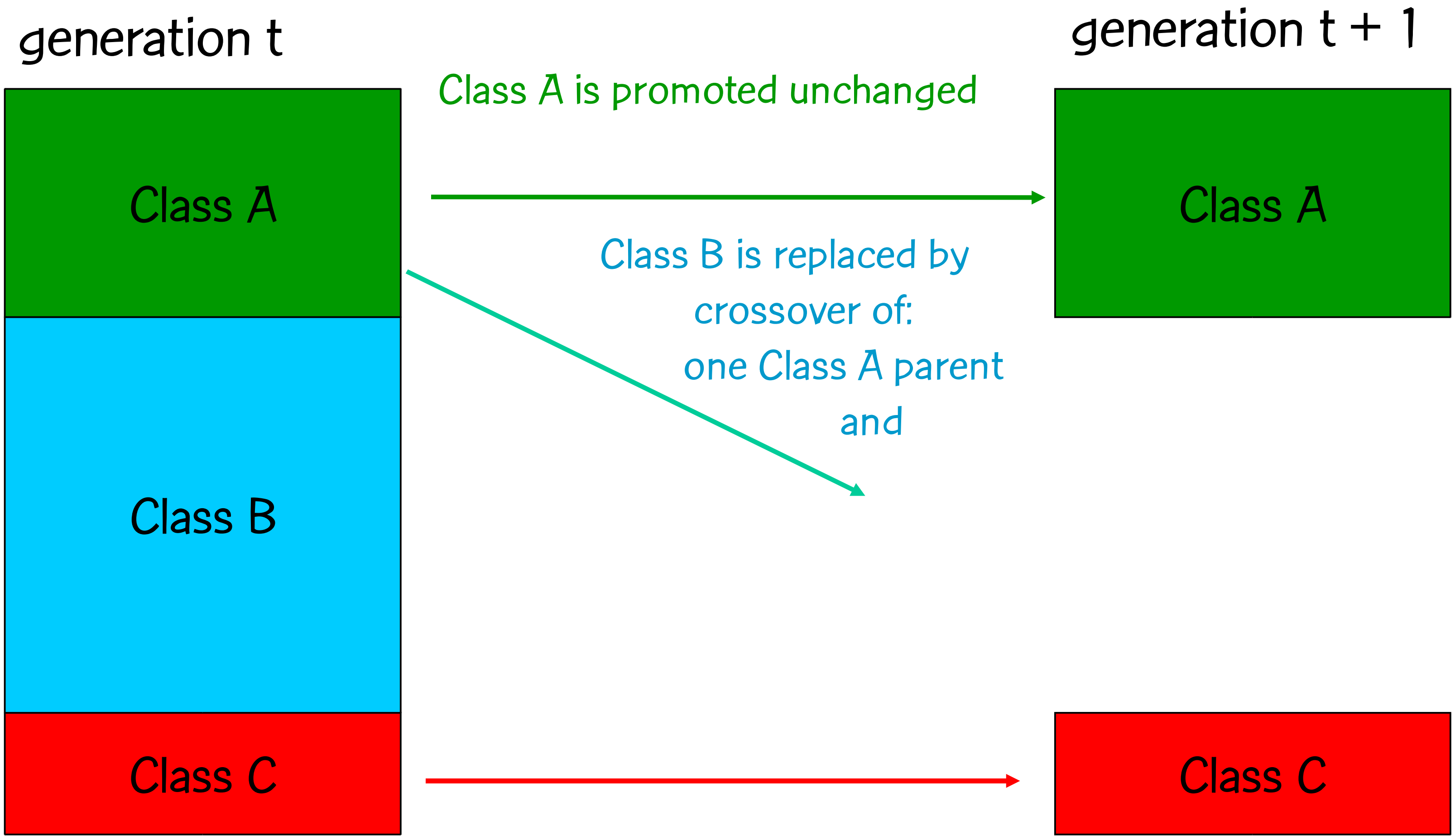
Population dynamics



Class A is promoted unchanged

Class C is replaced by randomly generated solutions.

Population dynamics



generation t

generation t + 1

Class A is promoted unchanged

Class A

Class A

Class B

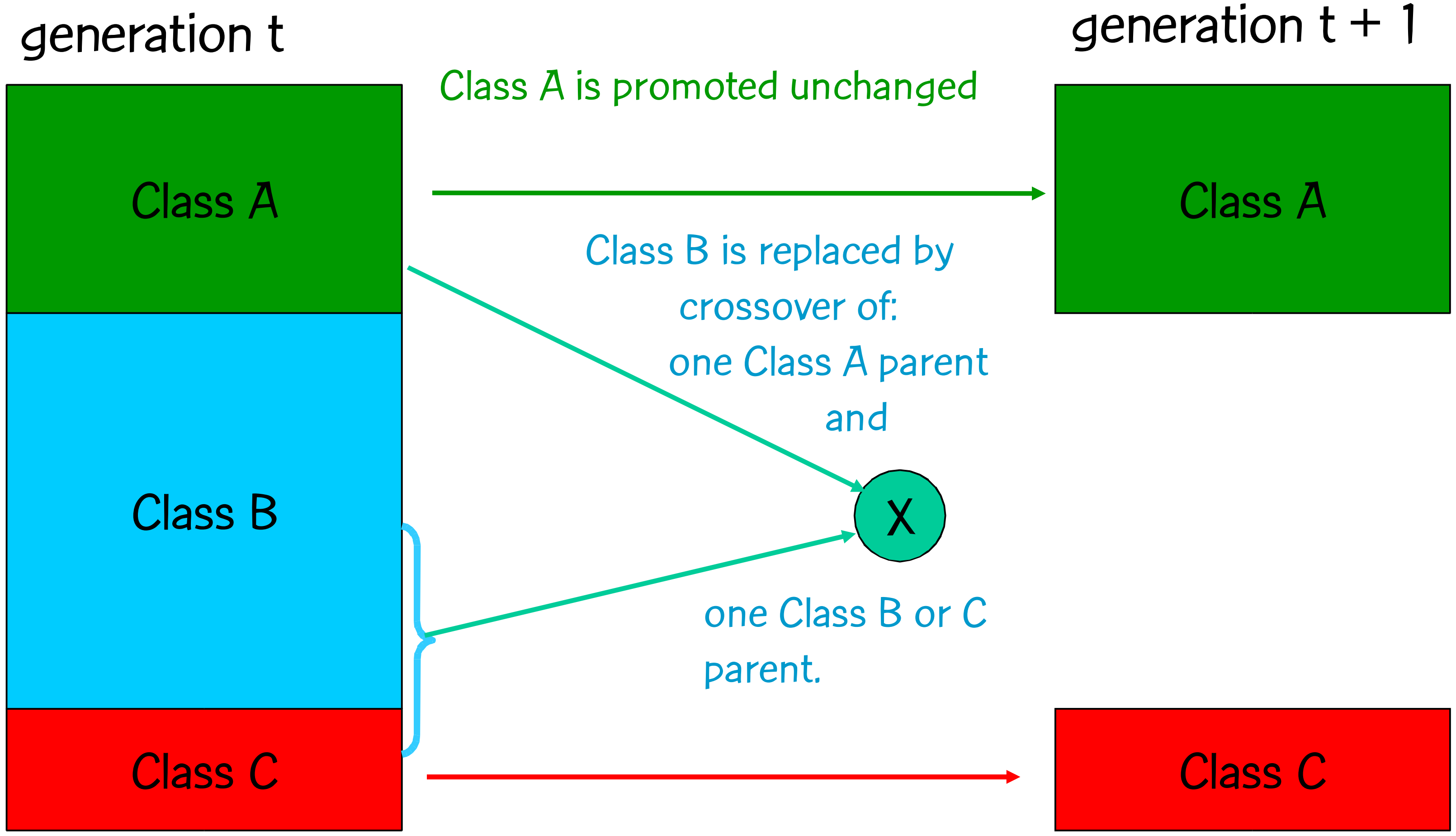
Class B is replaced by crossover of: one Class A parent and

Class C

Class C

Class C is replaced by randomly generated solutions.

Population dynamics

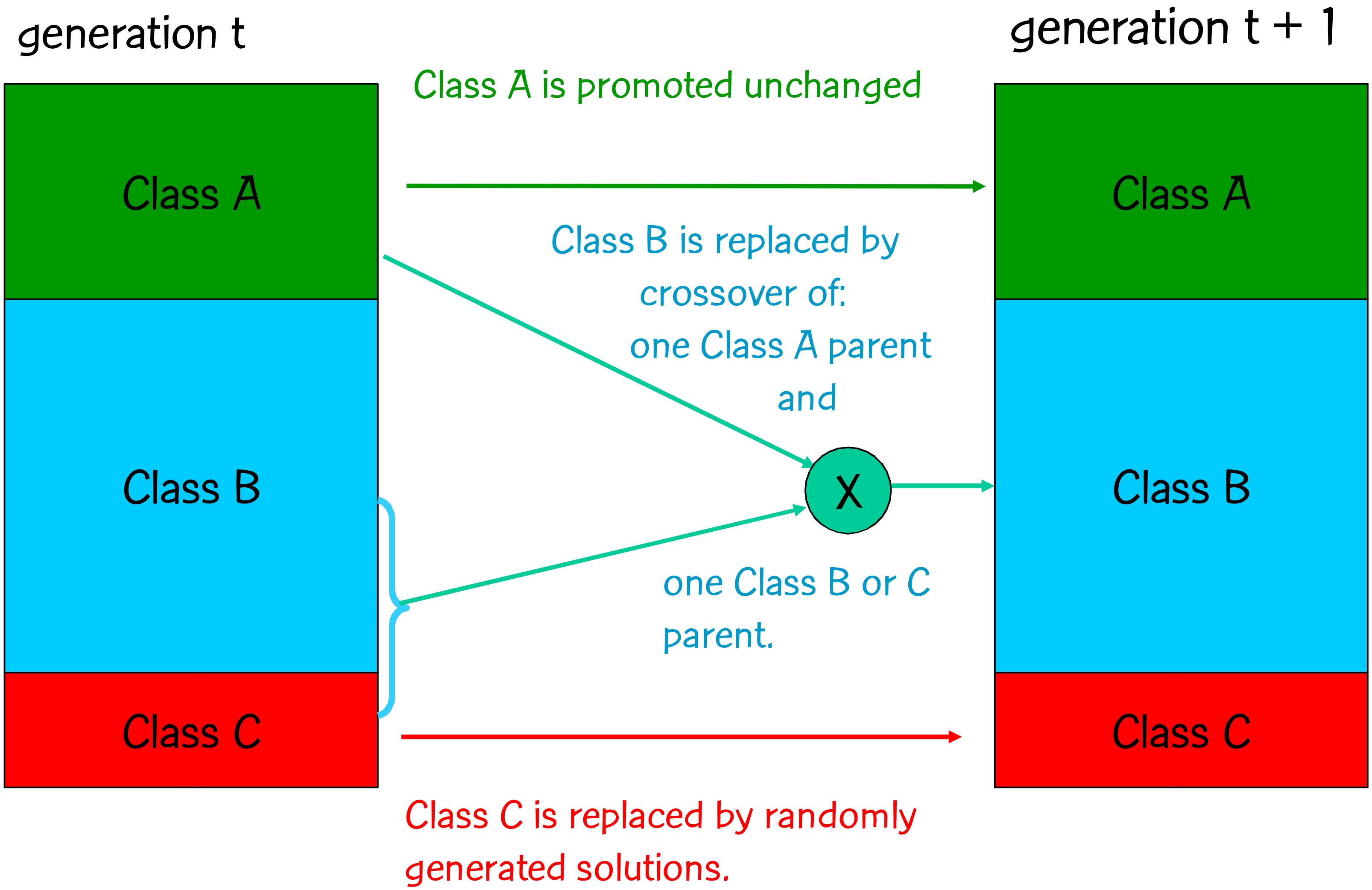


Class A is promoted unchanged

Class B is replaced by crossover of: one Class A parent and one Class B or C parent.

Class C is replaced by randomly generated solutions.

Population dynamics



generation t

generation t + 1

Class A is promoted unchanged

Class A

Class B

Class C

Class A

Class B

Class C

Class B is replaced by crossover of:
one Class A parent
and

X

one Class B or C parent.

Class C is replaced by randomly generated solutions.

Parent selection

- Parents are chosen at random:
  - one parent from Class A (elite).
  - one parent from Class B or C (non-elite).
- Reselection is allowed, i.e. parents can breed more than once per generation.
- Better individuals are more likely to reproduce.

Crossover with random keys

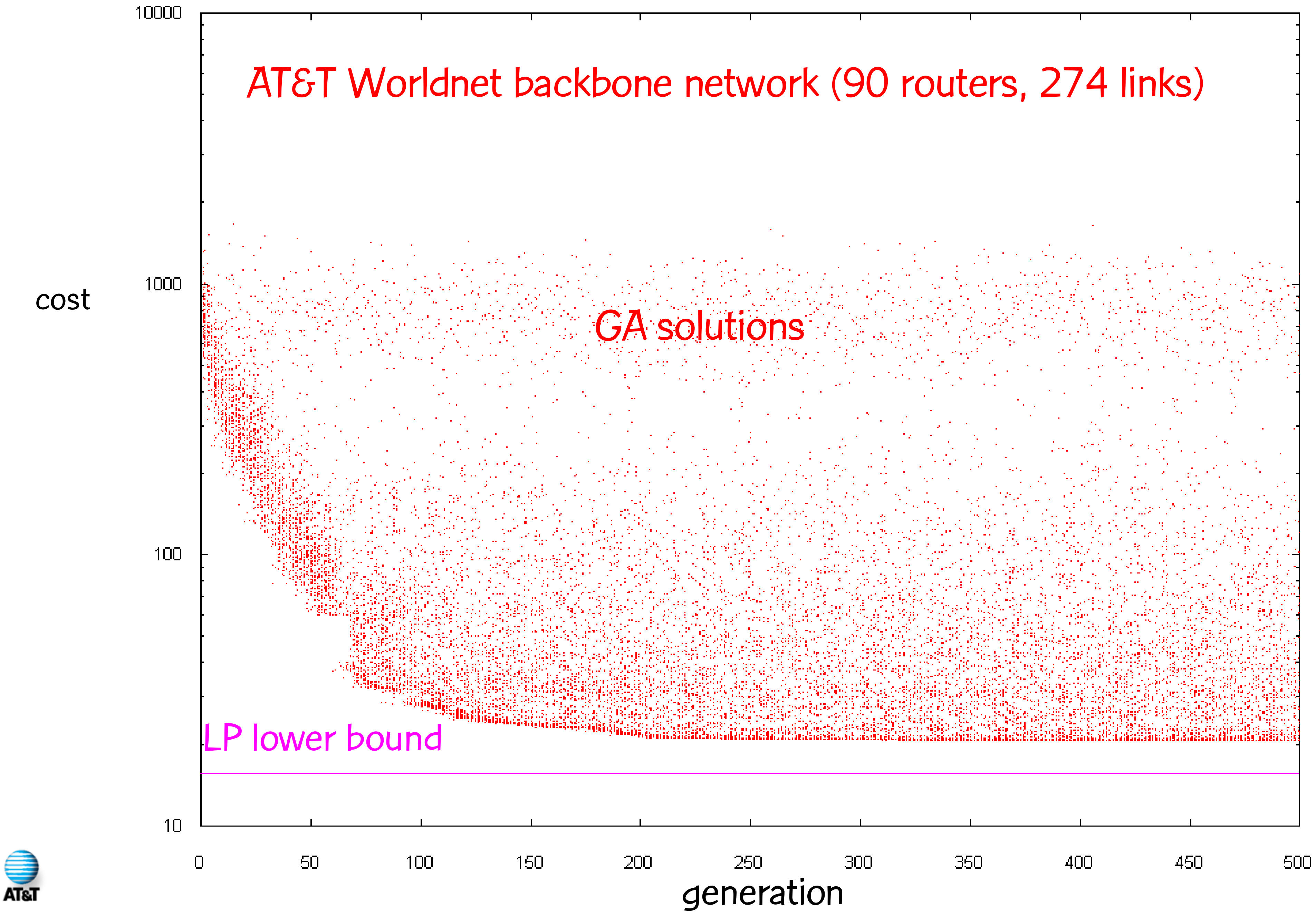
Bean (1994)

Crossover combines elite parent p_1 with non-elite parent p_2 to produce child c :

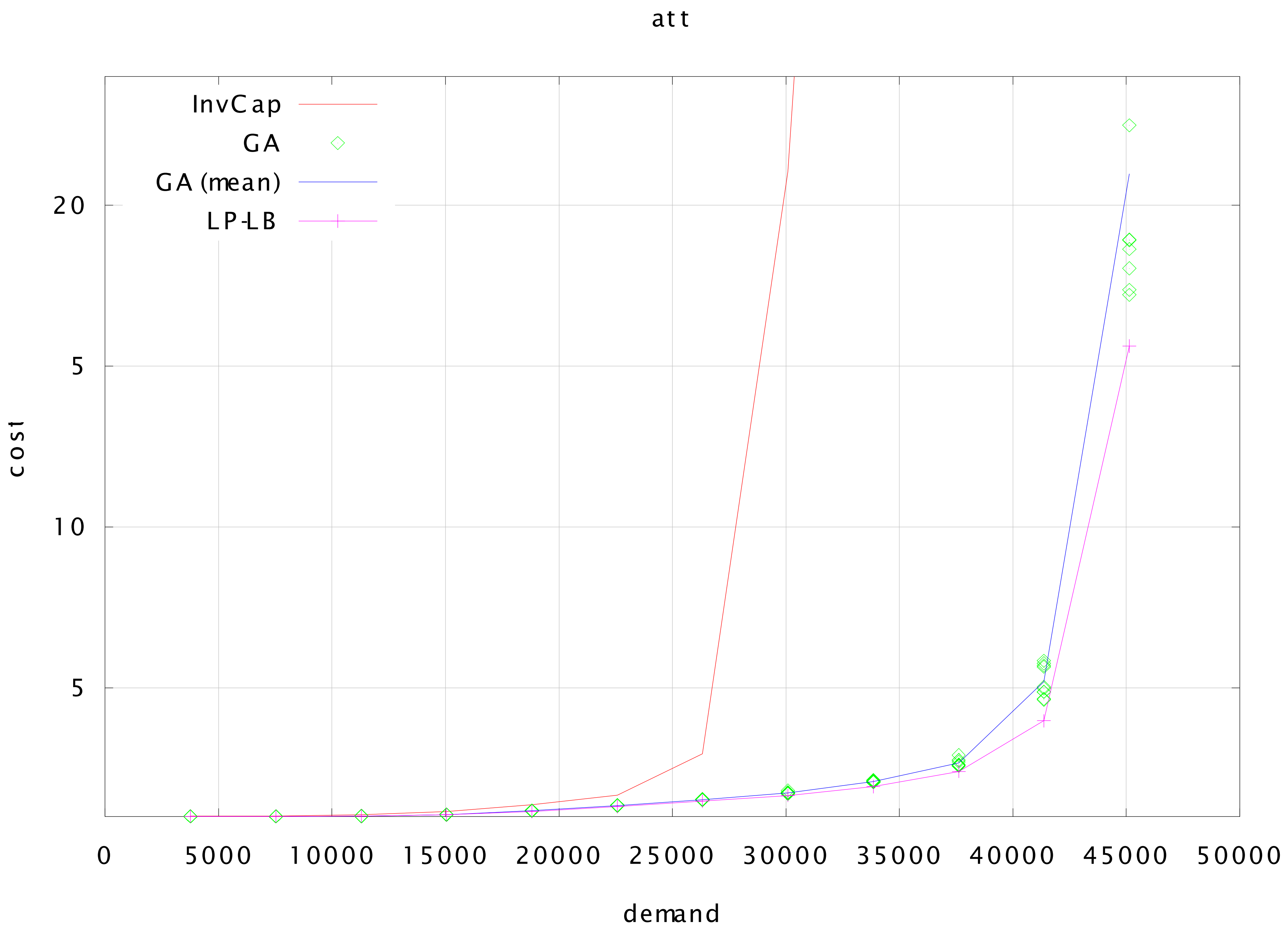
```
for all genes $i = 1,2,\ldots,|\bar{A}|$ do
    if $rrandom[0,1] < 0.01$ then
        $c[i] = irandom[1, w_{max}]$
    else if $rrandom[0,1] < 0.7$ then
        $c[i] = p_1[i]$
    else $c[i] = p_2[i]$
end
```

With small probability child has single gene mutation.

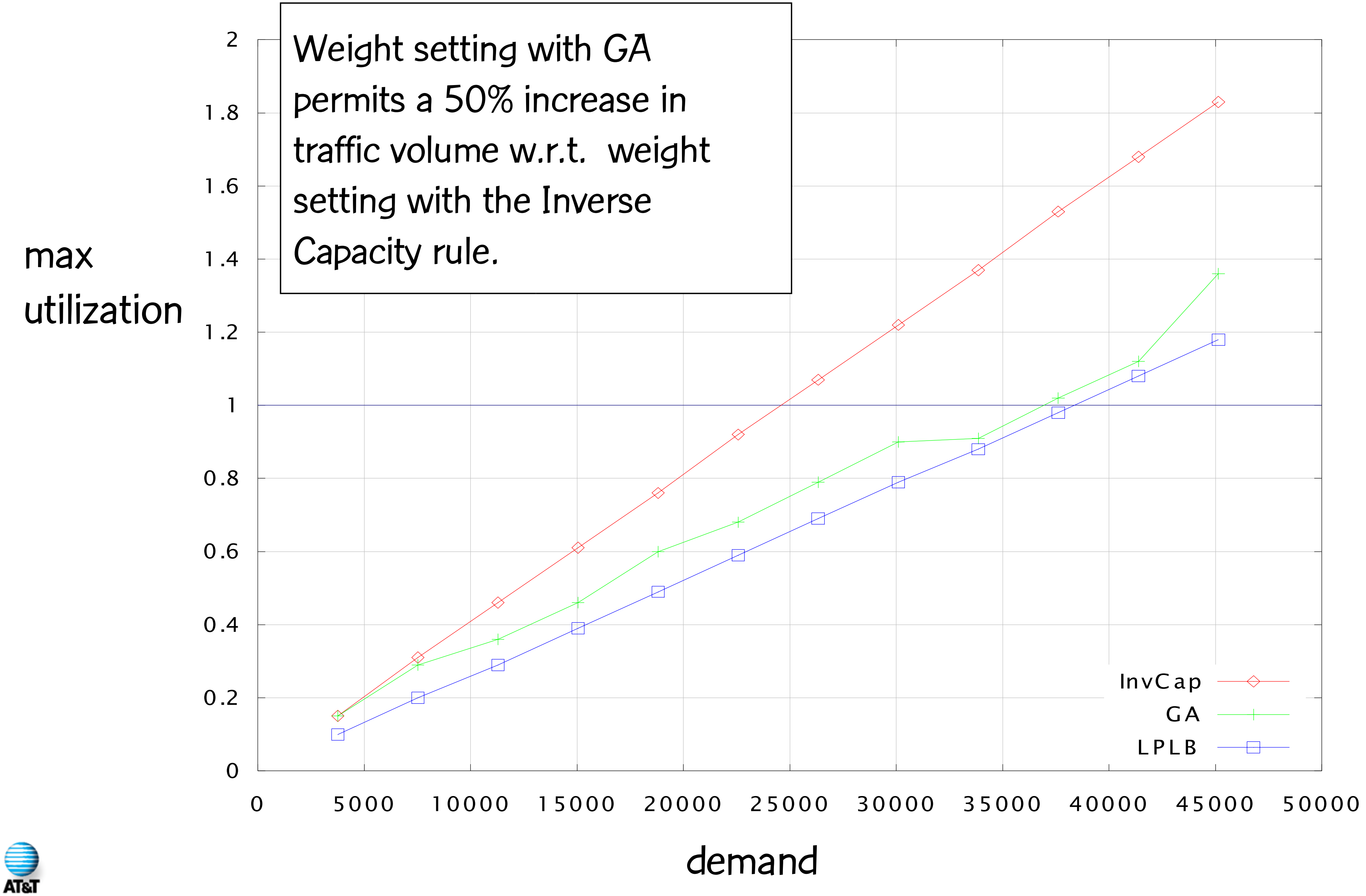
Child is more likely to inherit gene of elite parent.


AT&T Worldnet backbone network (90 routers, 274 links)
GA solutions
LP lower bound
cost
10000
1000
100
10
0
50
100
150
200
250
300
350
400
450
500
generation

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

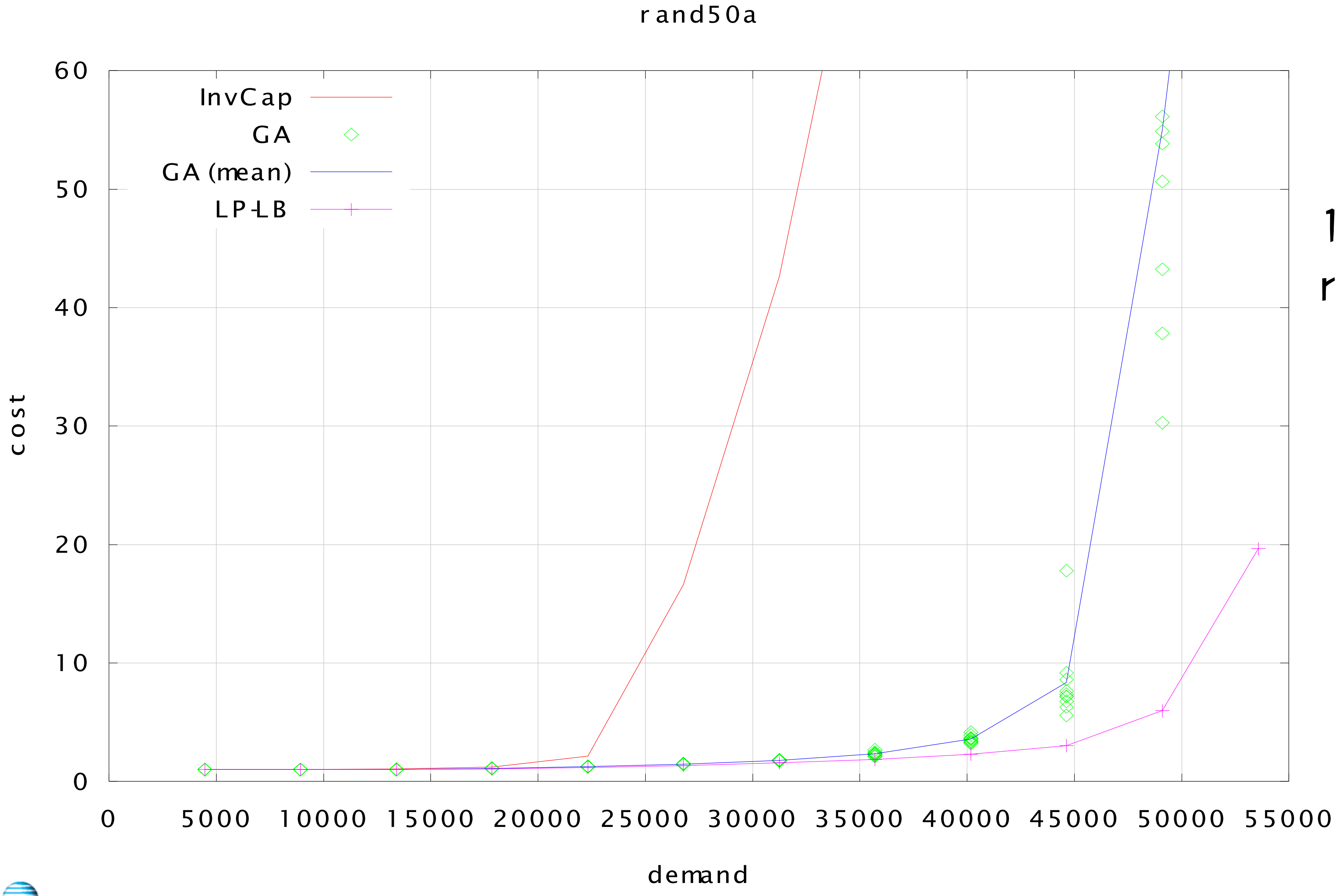


AT&T Worldnet backbone network (90 routers, 274 links)



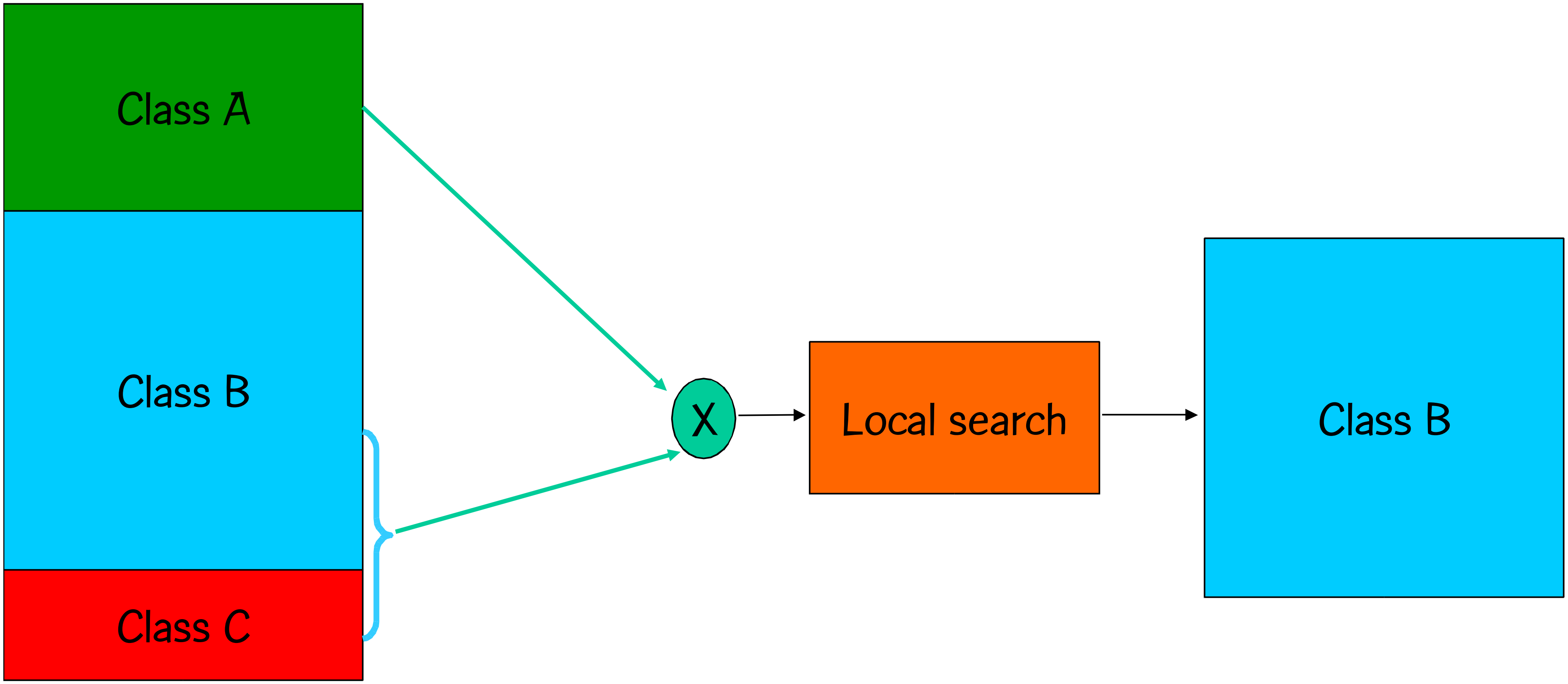
Weight setting with GA permits a 50% increase in traffic volume w.r.t. weight setting with the Inverse Capacity rule.

Rand50a: random graph with 50 nodes and 245 arcs.



1 hour run

Optimized crossover = crossover + local search



Fast local search

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

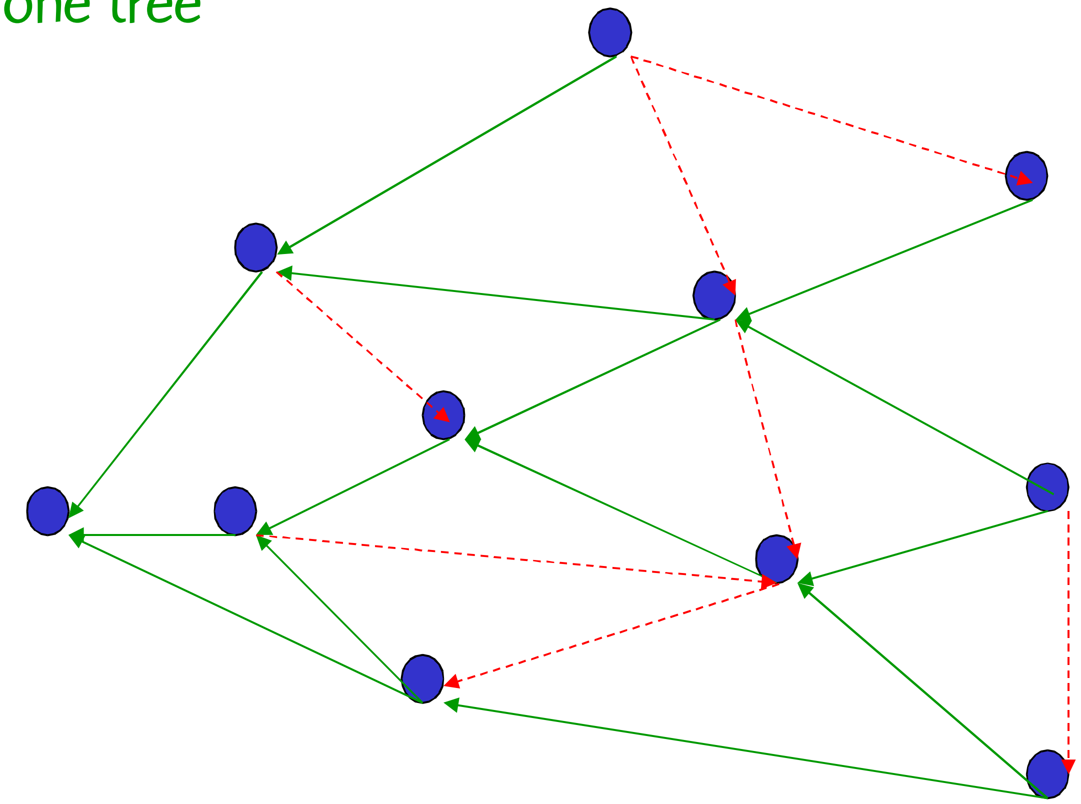
Dynamic shortest path

- In local search, when arc weight increases, shortest path trees:
  - may change completely (rarely do)
  - may remain unchanged (e.g. arc not in a tree)
  - may change partially
    - Few trees change
    - Small portion of tree changes

} Does not make sense to recompute trees from scratch.

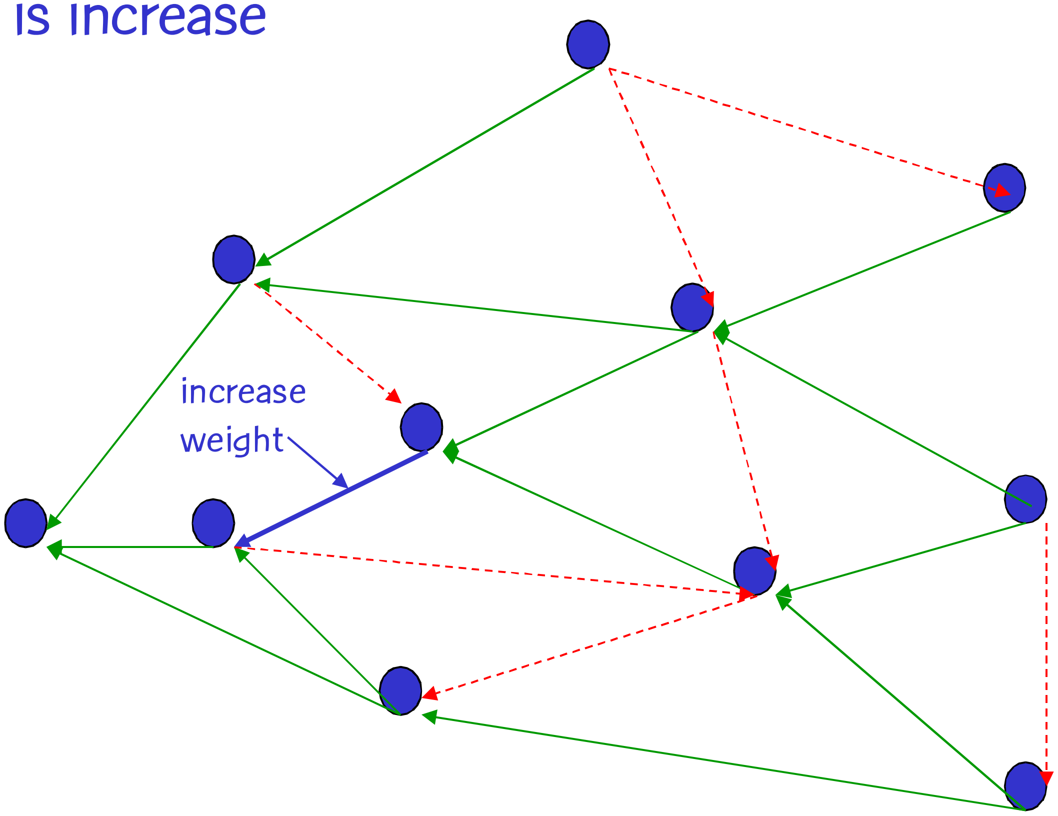
Dynamic shortest path

Consider one tree
at a time.



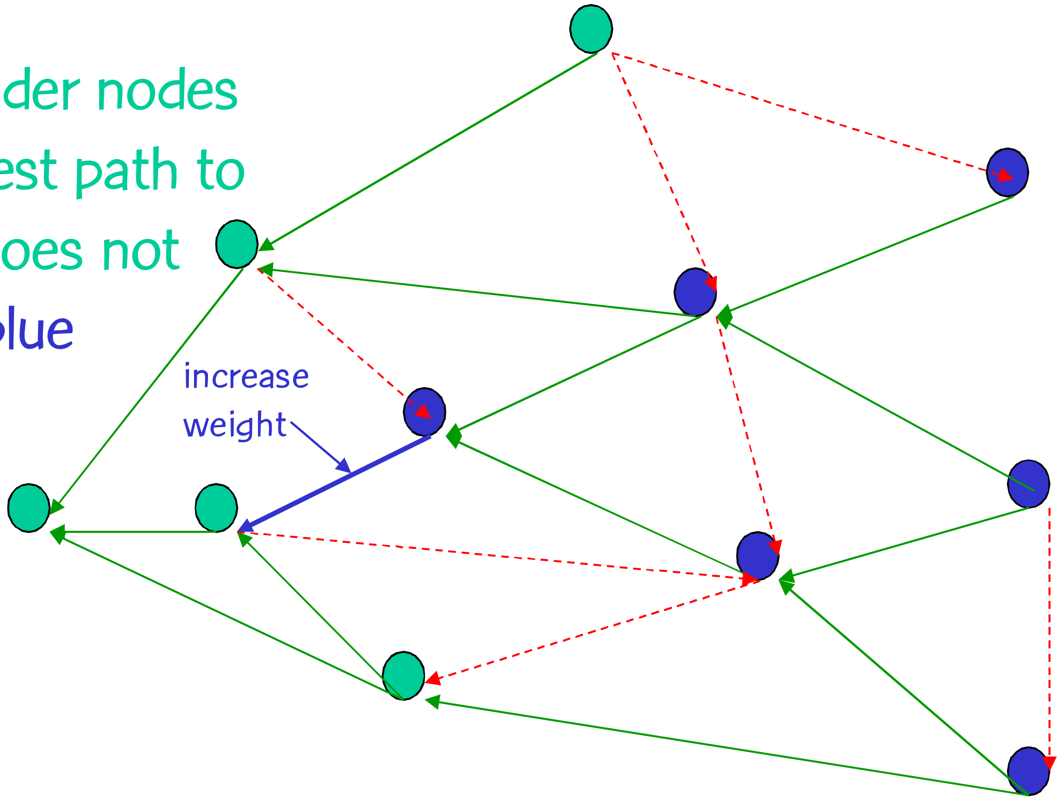
Dynamic shortest path

Arc weight is increase by 1.

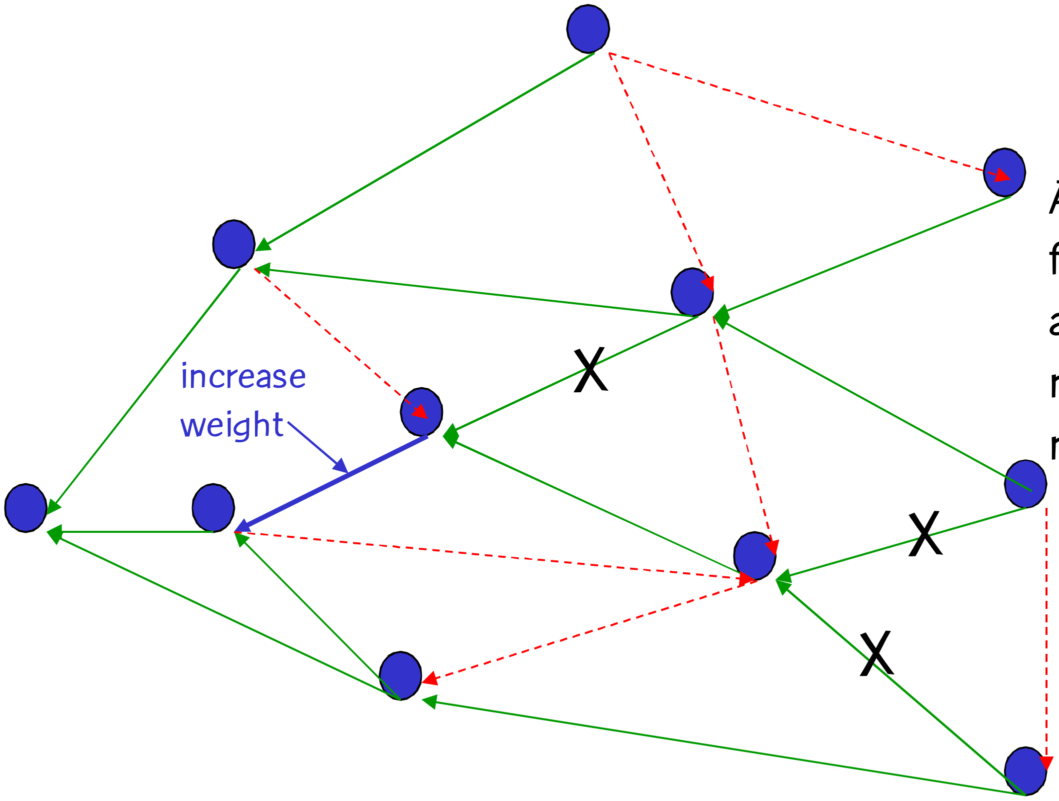


Dynamic shortest path

Do not consider nodes whose shortest path to destination does not go through blue arc.

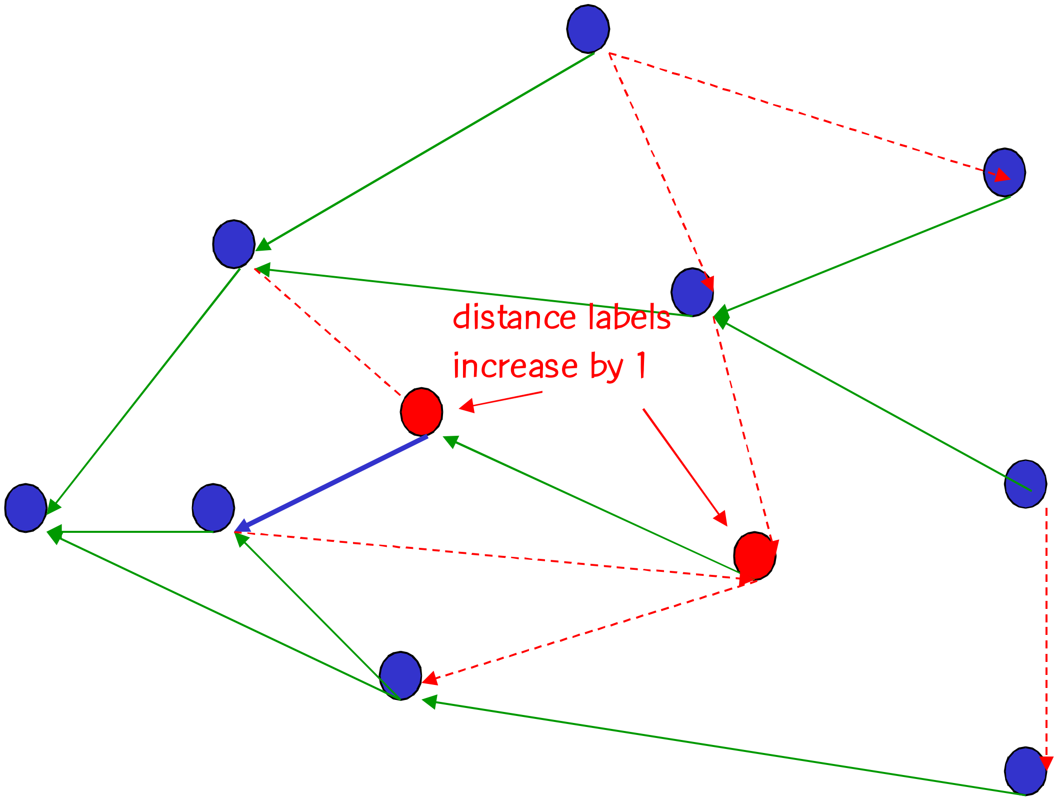


Dynamic shortest path



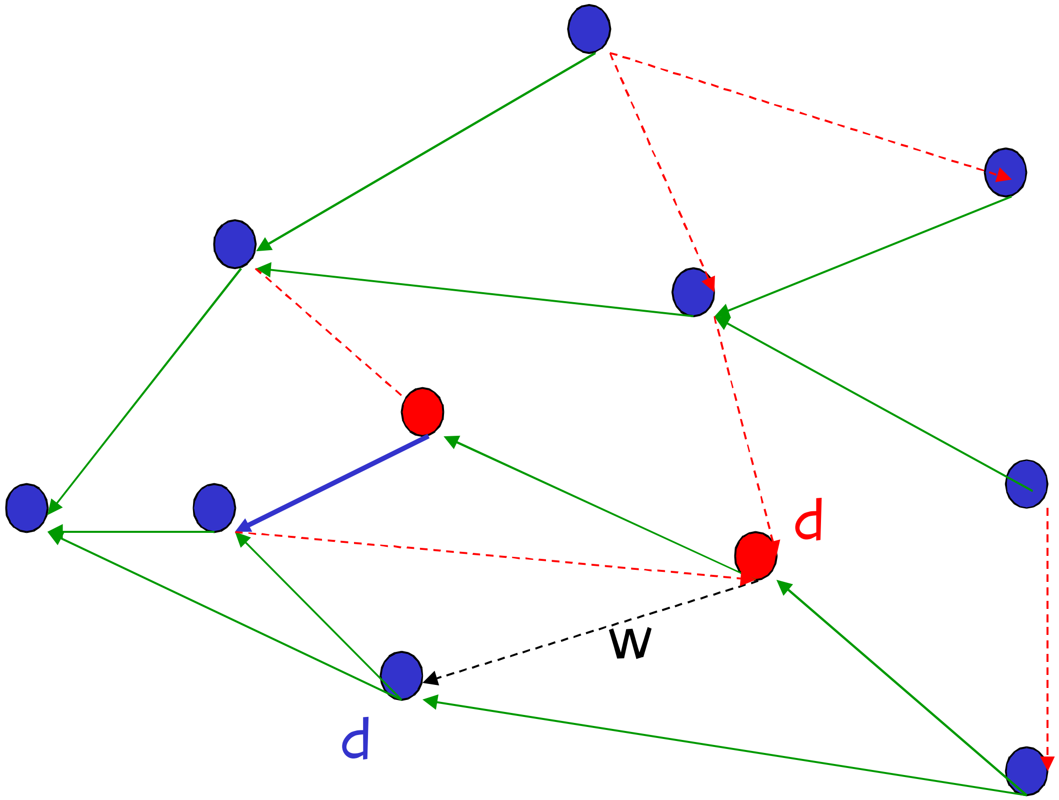
Arc (u,v) is removed from tree since alternative paths from node u to the destination node exist.

Dynamic shortest path



Shortest paths
from red nodes
must traverse
blue arc.

Dynamic shortest path

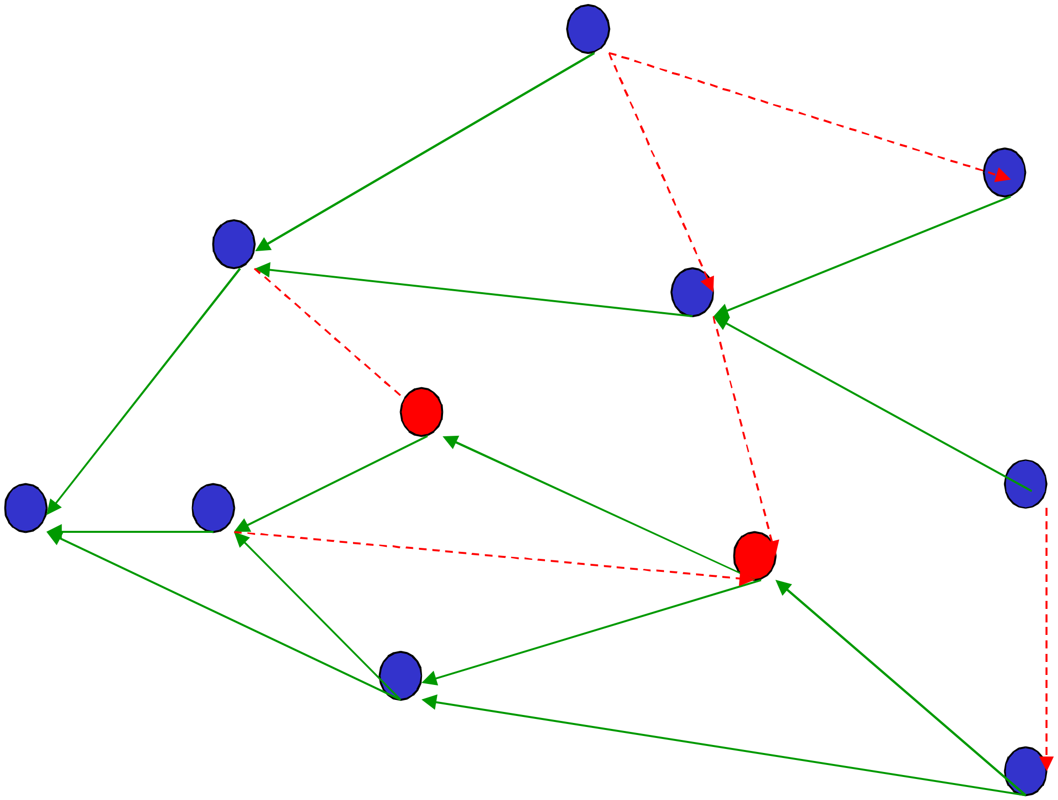


Test all arcs of type

🔴 → 🔵

If d – d = w , then 🔴 → 🔵 enters tree.

Dynamic shortest path



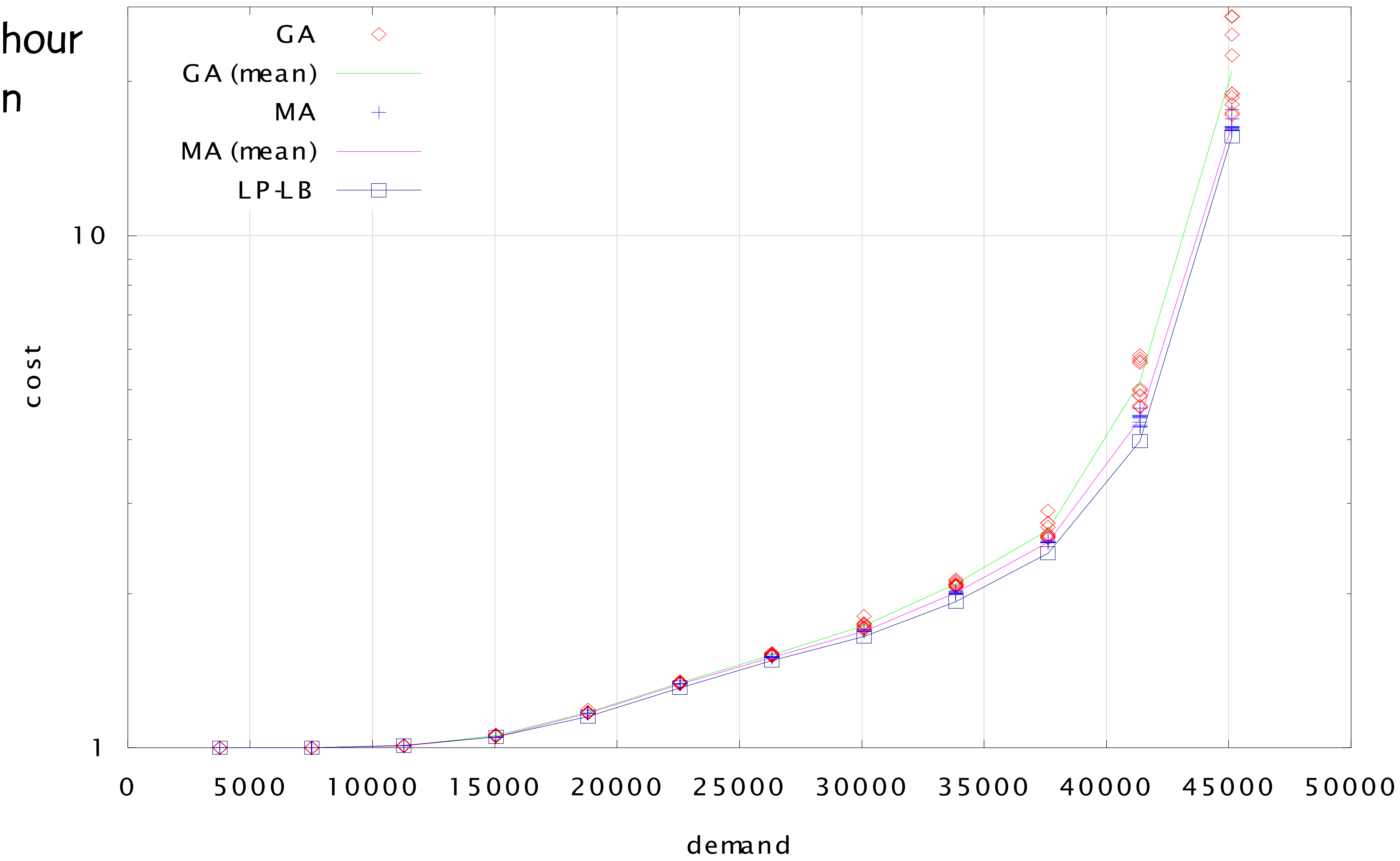
Dynamic shortest path

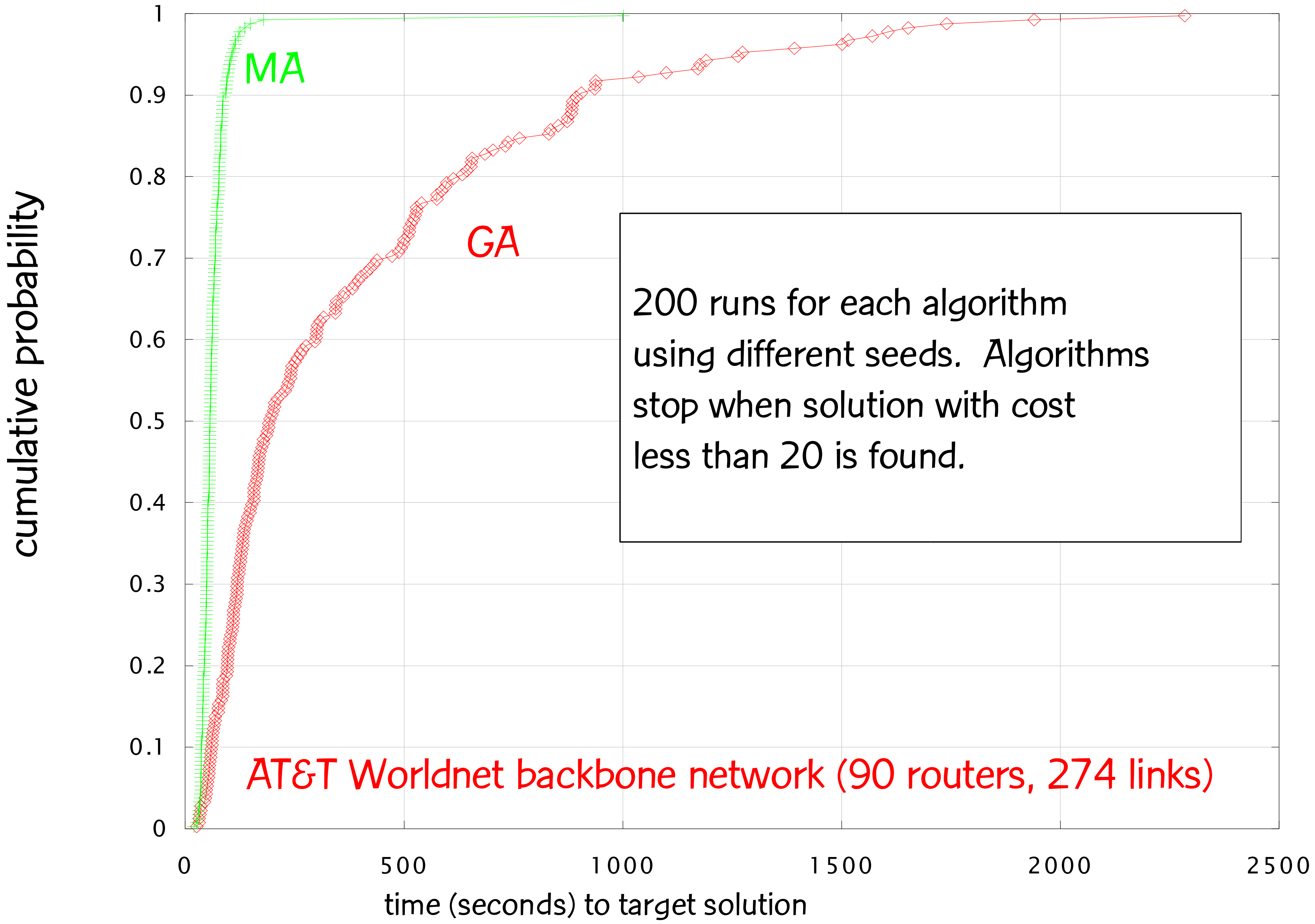
L.S. Buriol, M.G.C. Resende, & M. Thorup, “Speeding up dynamic shortest path algorithms,” AT&T Labs Research Report, 2003.

- Ramalingam & Reps (1996) allow arbitrary arc weight change.
- We specialized the Ramalingam & Reps algorithm for unit arc weight change.
  - Avoid use of heaps
  - Achieve a factor of 2 |5 speedup w.r.t. Ramalingam & Reps on these test problems

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

1 hour run



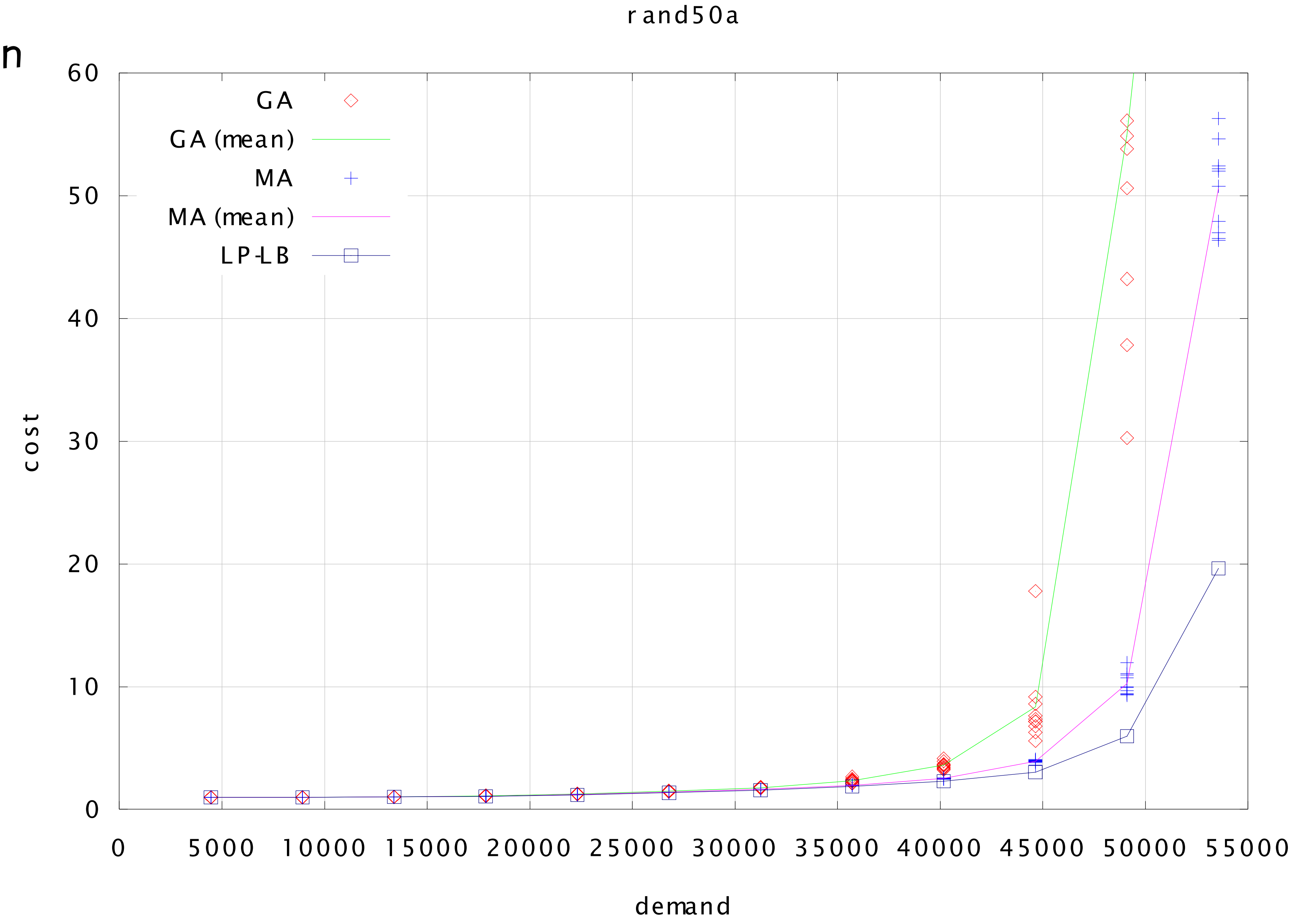


200 runs for each algorithm using different seeds. Algorithms stop when solution with cost less than 20 is found.

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

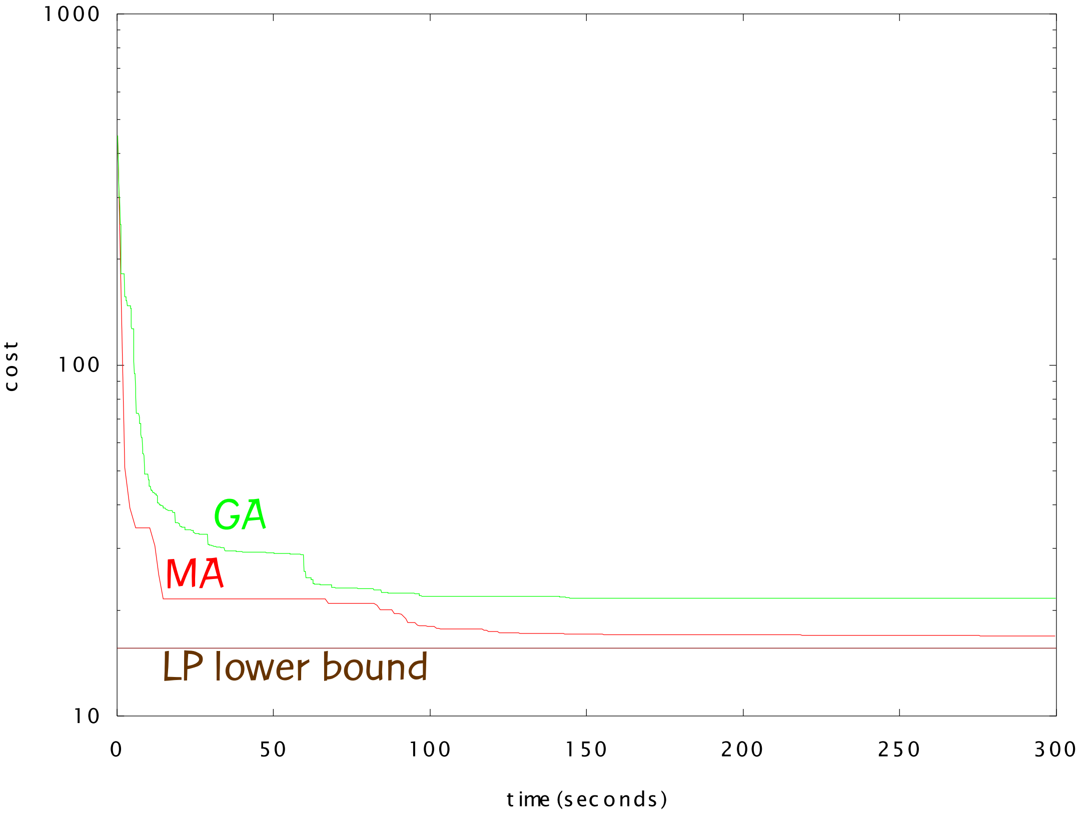
Rand50a: random graph with 50 nodes and 245 arcs.

1 hour run



Remark

- Memetic algorithm (MA) improves over pure genetic algorithm (GA) in two ways:
  - Finds solutions faster
  - Finds better solutions



Application 6:

Survivable IP network design

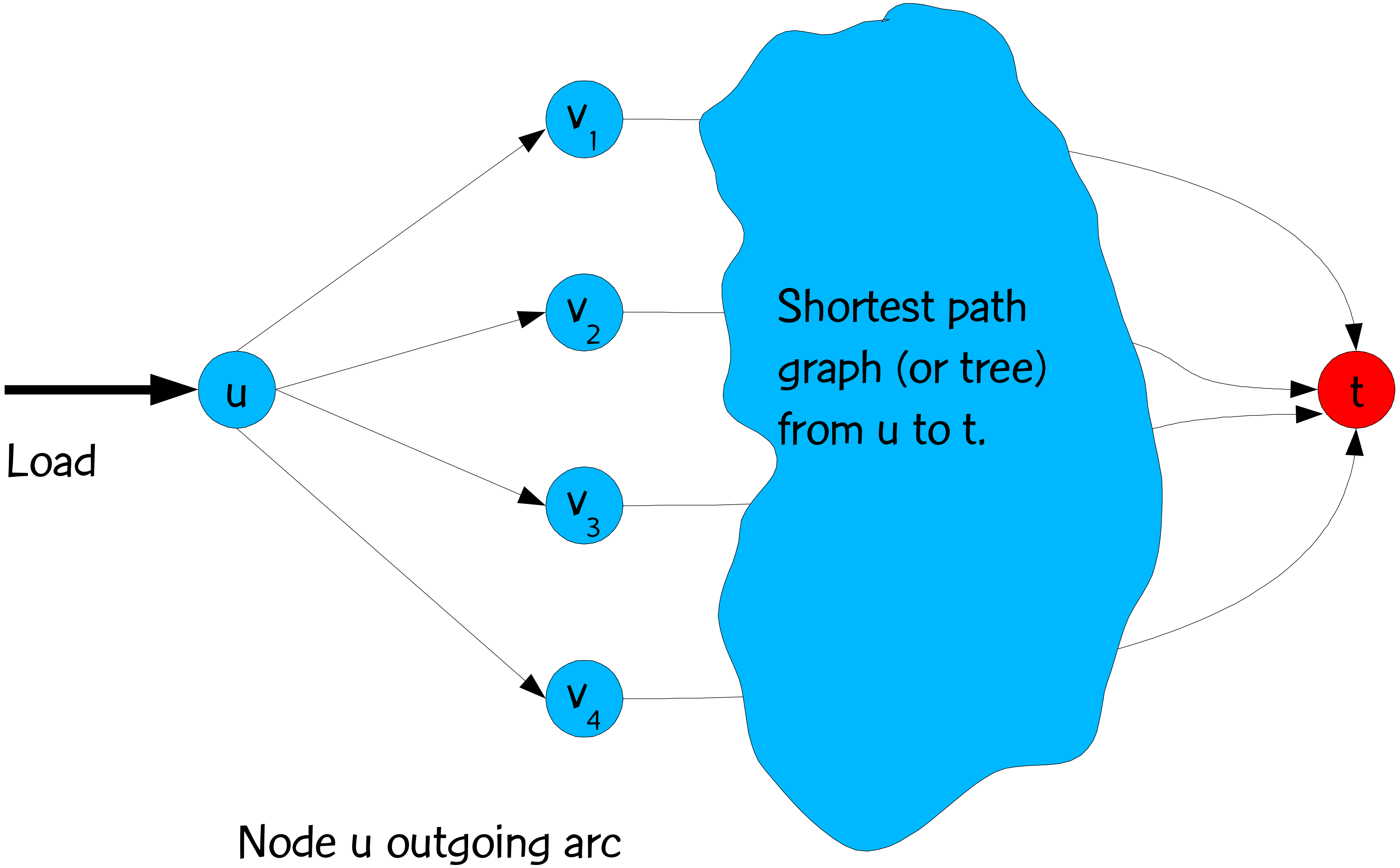
Survivable IP network design

- Given
  - $G = (N,\mathcal{A})$, where:
    - N is the set of routers
    - $\mathcal{A}$ is the set of potential arcs where capacity can be installed.
  - Demand matrix D = [d], such that for each (u,v) | N · N
    - d(u,v) is the traffic demand from router u to router v.
  - Single link capacity M

Survivable IP network design

- Determine, for each arc a
  - OSPF weight $w_a \in [1, w_{max}]$
  - Number of links of capacity M installed in arc a (arc multiplicity)
- Such that
  - There is sufficient capacity to route all of the demand
  - Using OSPF routing with traffic splitting
  - Subject to single router or single arc failure

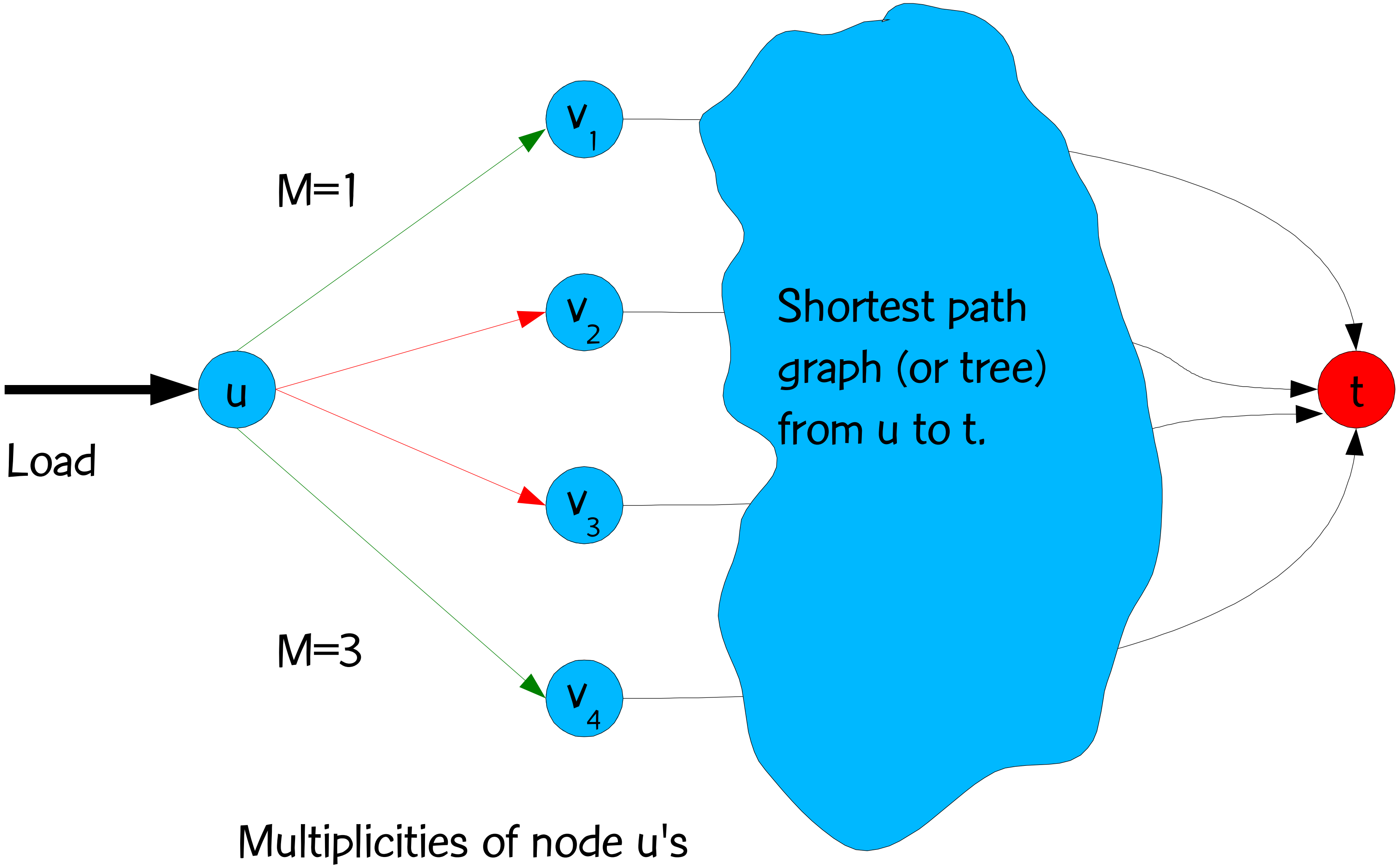
Traffic splitting



Shortest path graph (or tree) from u to t.

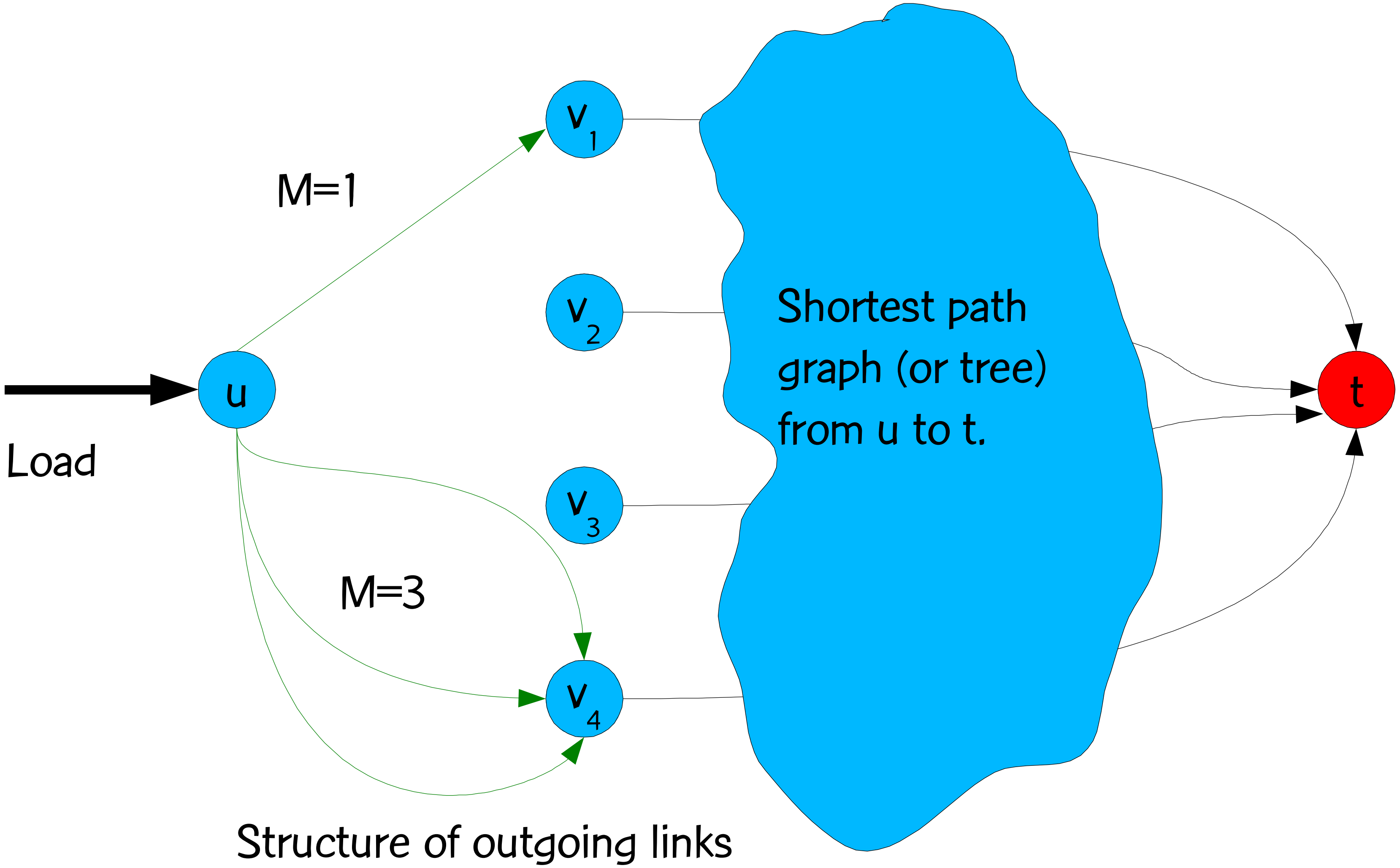
Node u outgoing arc structure

Traffic splitting



Multiplicities of node u's outgoing arcs

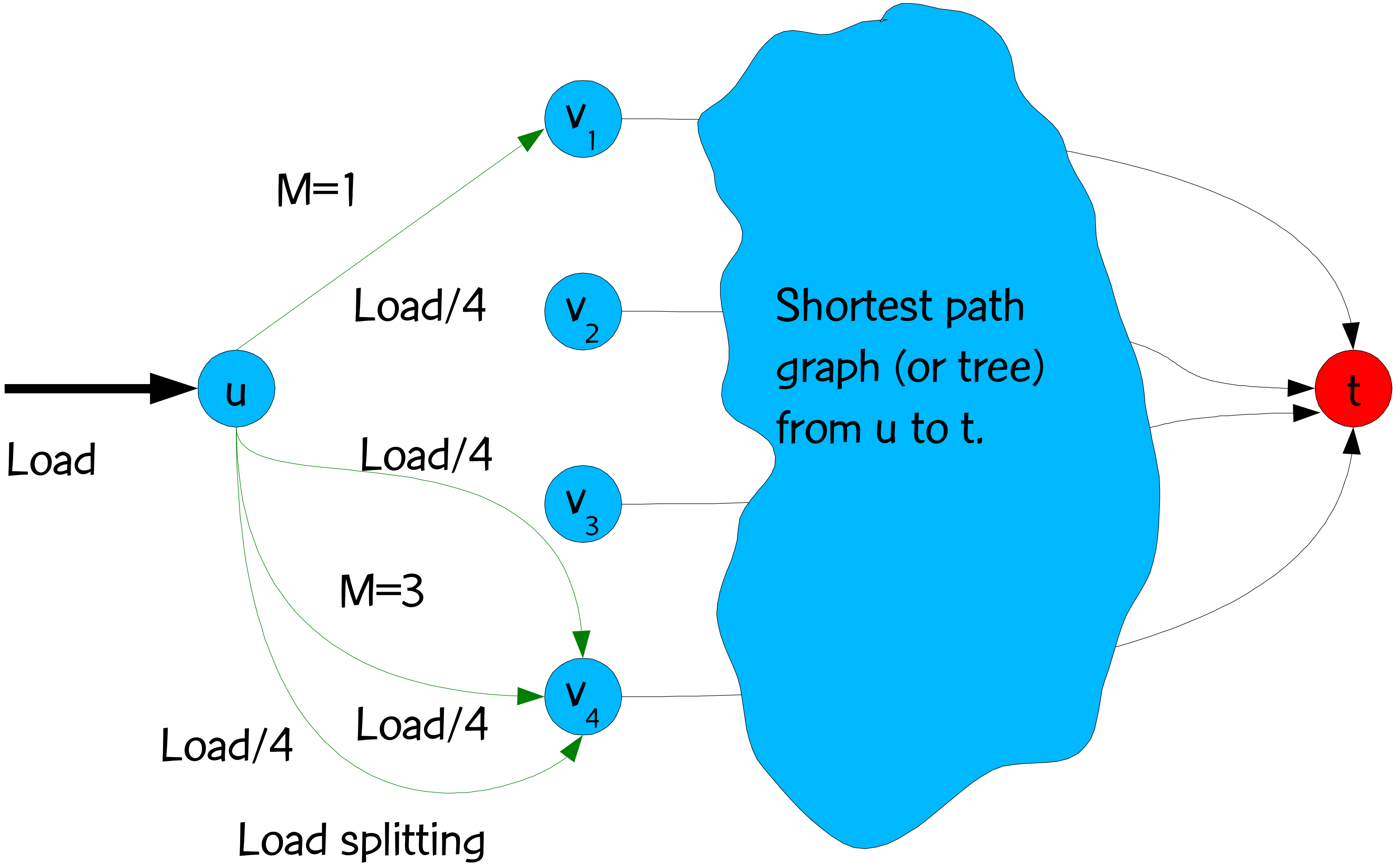
Traffic splitting



Shortest path graph (or tree) from u to t.

Structure of outgoing links of node u

Traffic splitting

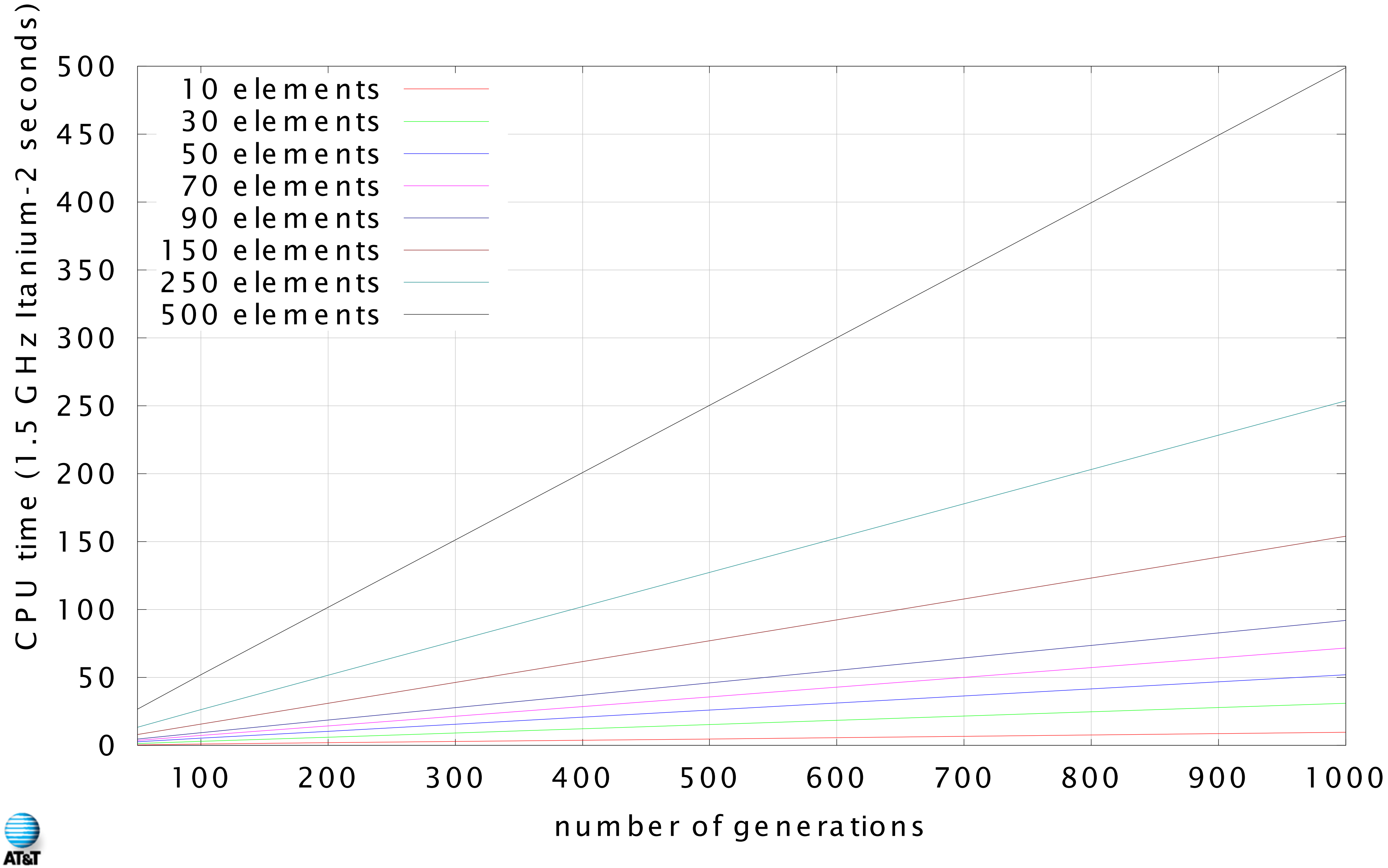


Load splitting

Genetic algorithm for no-failure case

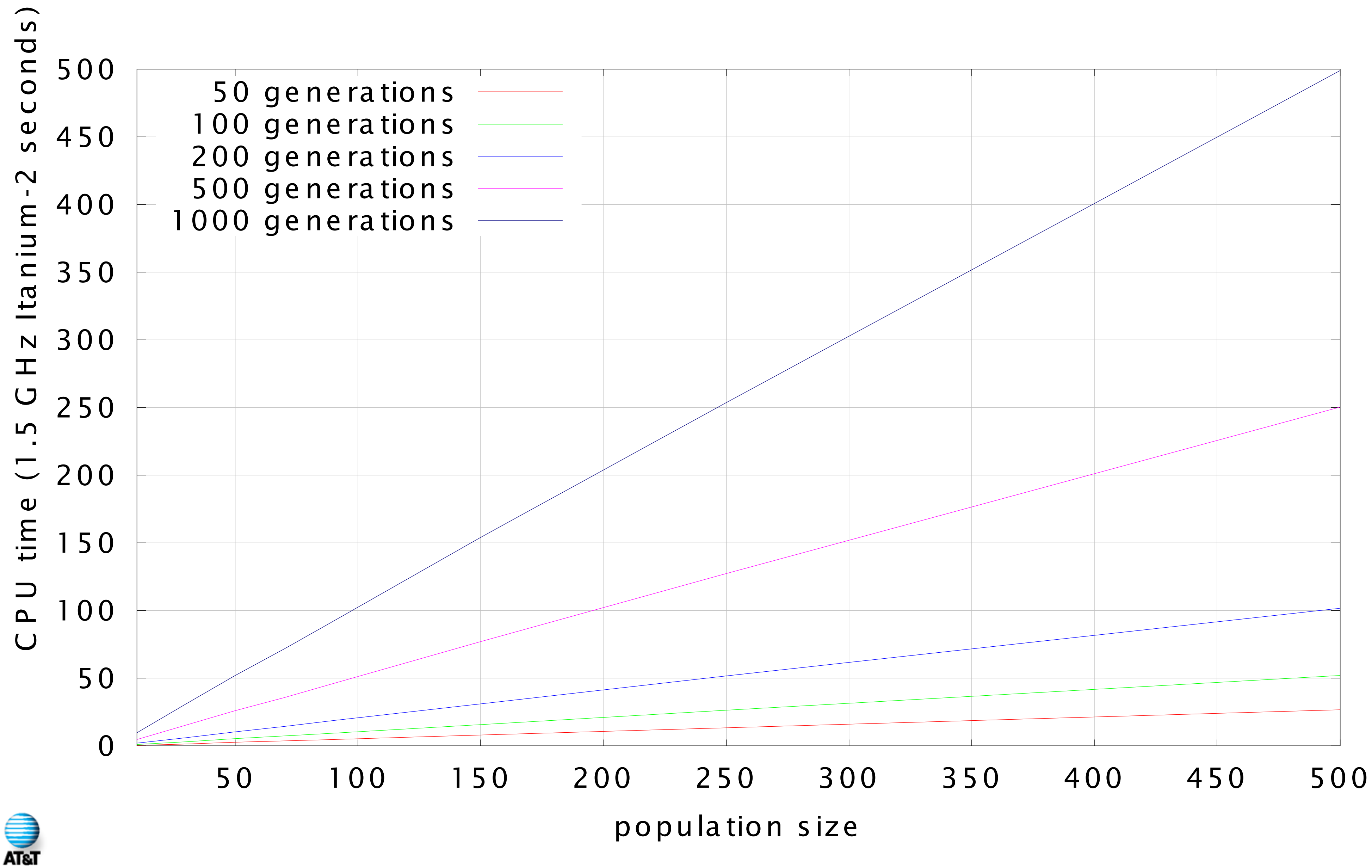
- Solutions are OSPF weight vectors.
- A OSPF weight vector defines shortest path graphs on which routing is done.
- Assume each arc has unit multiplicity.
- Repeat until feasible capacity/load is achieved:
  - Route demand and determine loads on arcs.
  - Determine arc multiplicities to insure minimum arc capacities required to flow loads on arcs. Multiplicities are never decreased.

Running time: 74-router, 278-arc, 18-terminal nodes, 306 demand pairs
No router or arc failure.

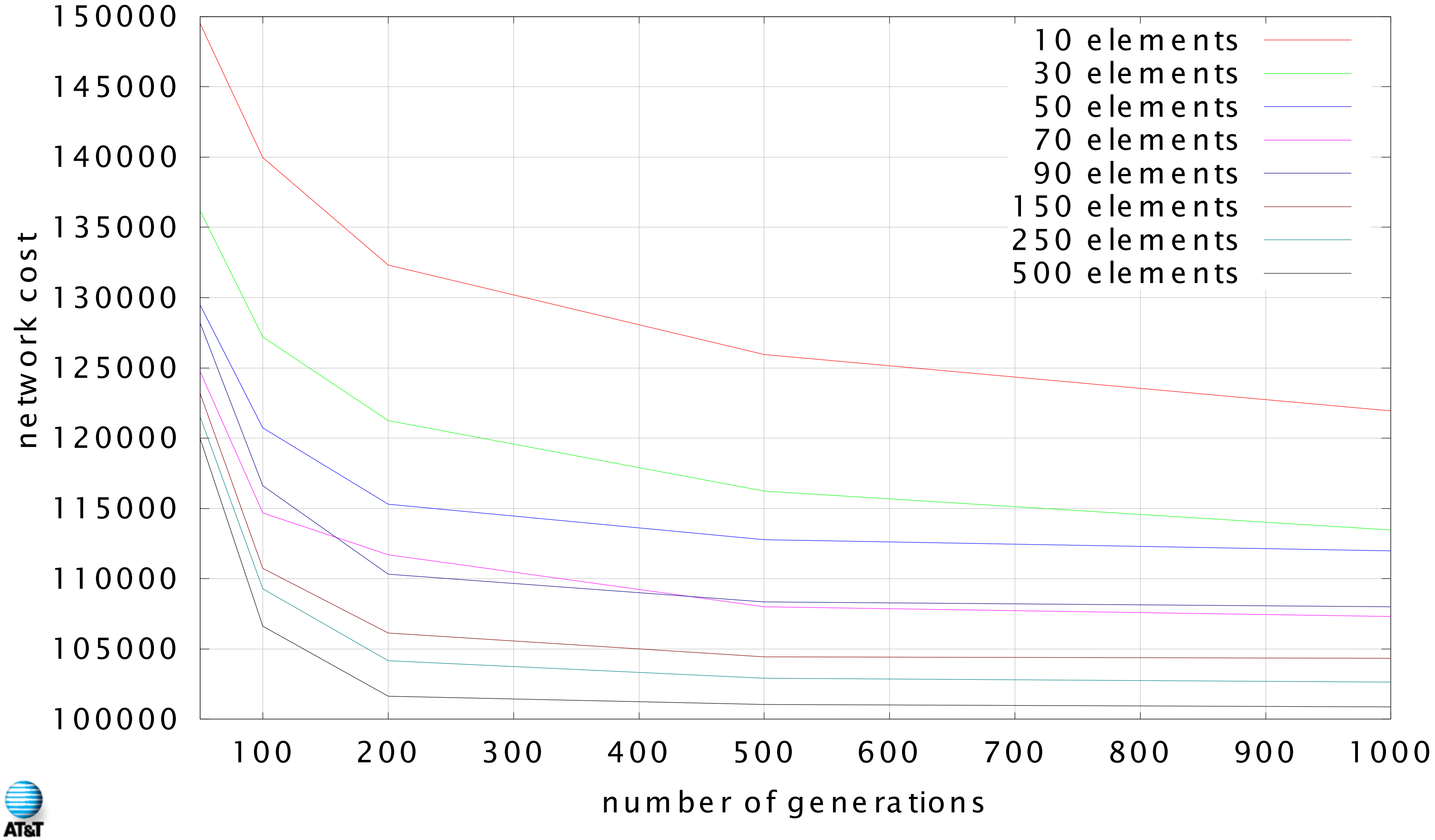


Running time: 74-router, 278-arc, 18-terminal nodes, 306 demand pairs

No router or arc failure.

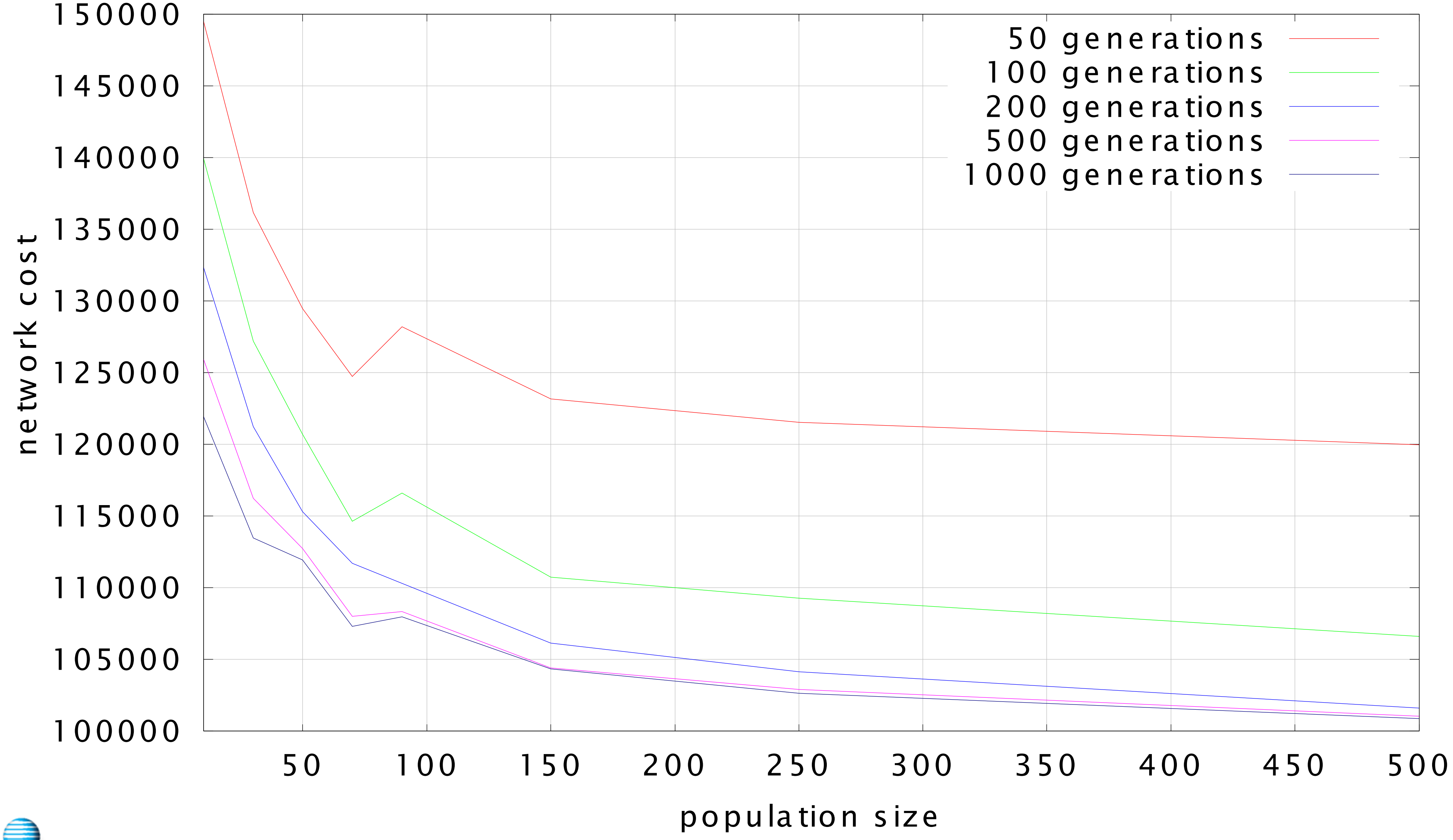


Network cost: 74-router, 278-arc, 18-terminal nodes, 306 demand pairs
No router or arc failure.



Network cost: 74-router, 278-arc, 18-terminal nodes, 306 demand pairs

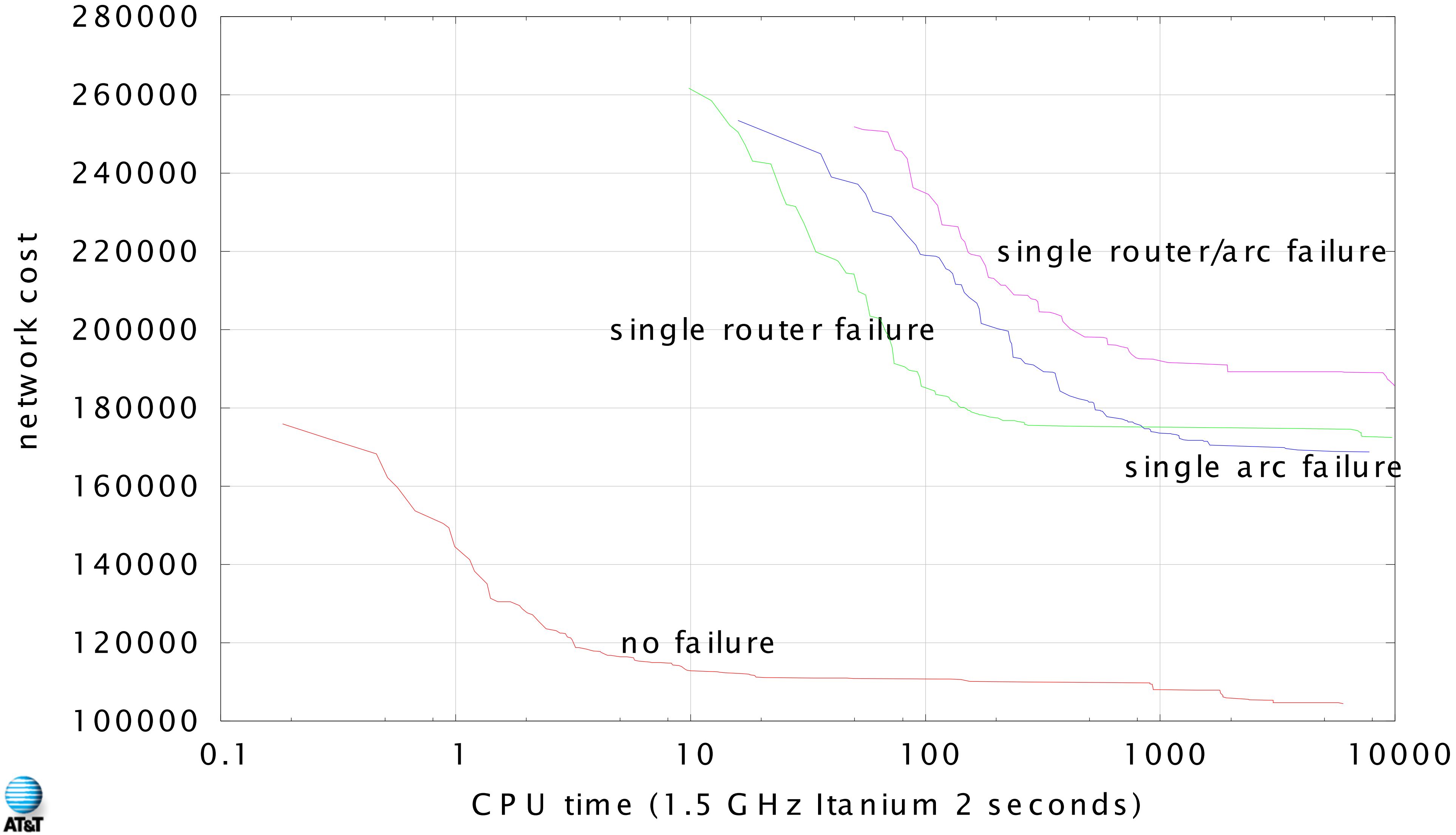
No router or arc failure.



Genetic algorithm for single-failure case

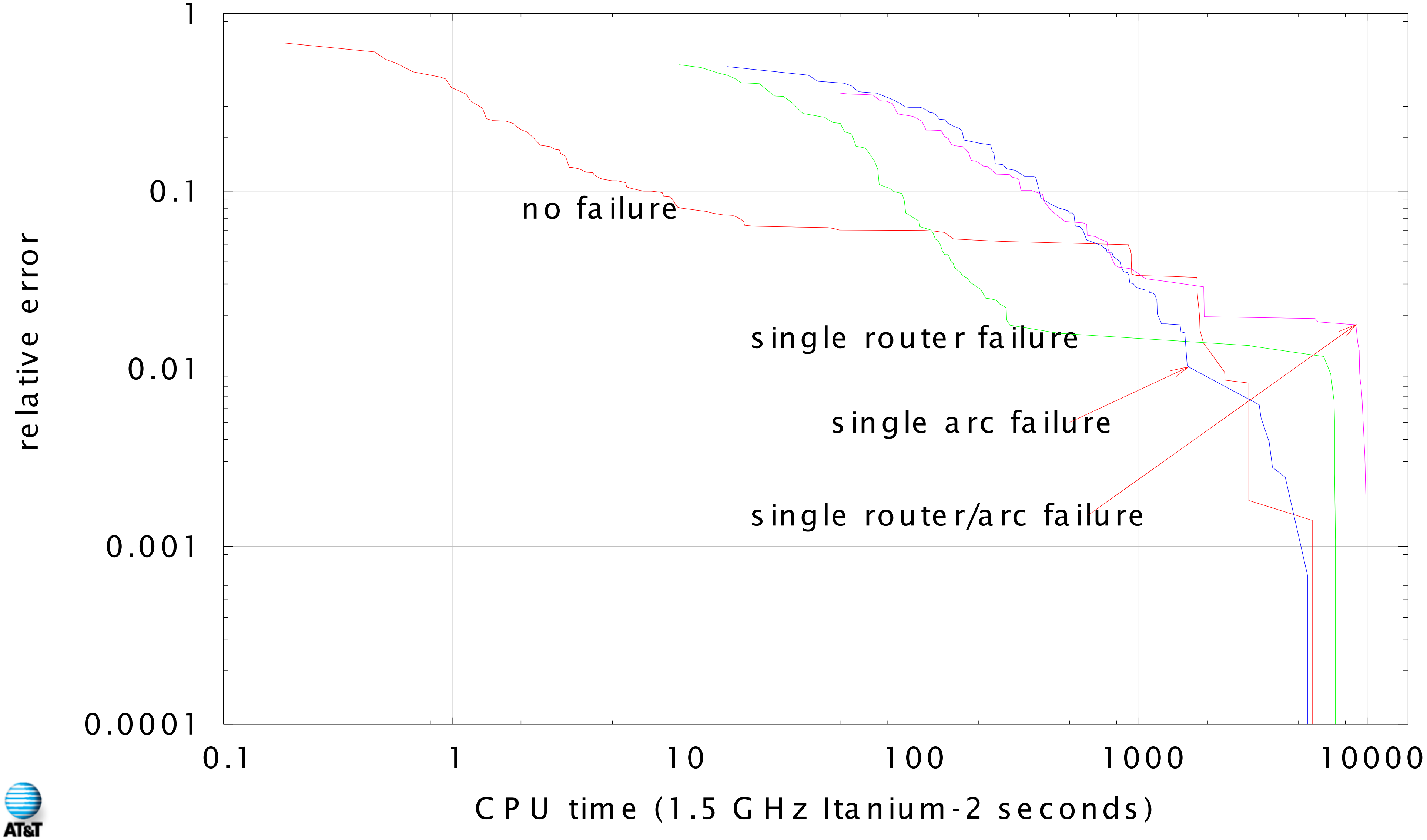
- Algorithm similar to no-failure case.
- Compute multiplicities for no-failure configuration and for each single-failure configuration.
- For each arc, set its multiplicity to be the maximum multiplicity over all simulated configurations.

Network cost: 74-router, 278-arc, 18-terminal nodes, 306 demand pairs
No router or arc failure, single-router failure, single-arc failure, and single-router or single-arc failure.

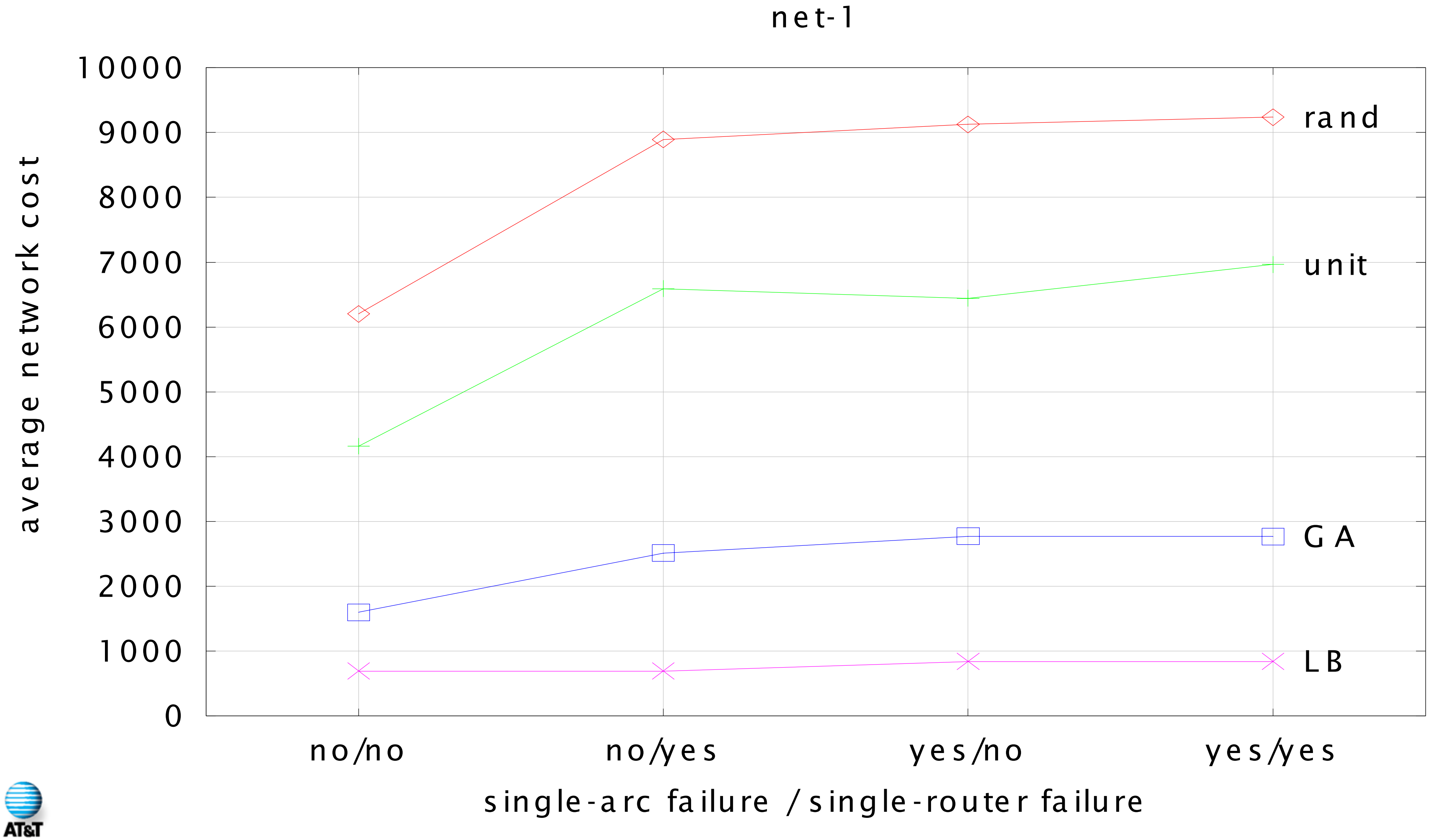


Relative error of network cost: 74-router, 278-arc, 18-terminal nodes, 306 demand pairs.

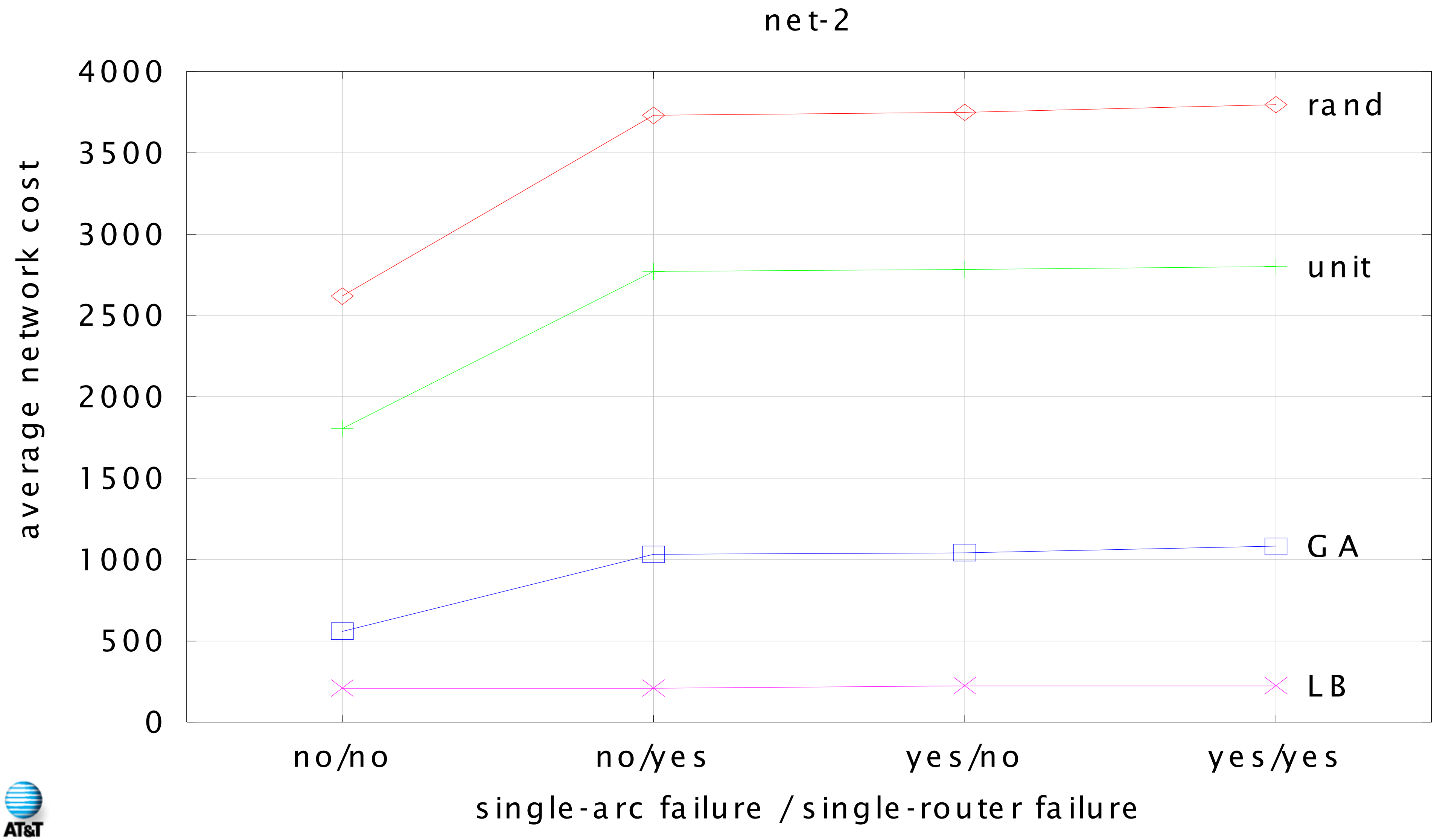
No router or arc failure, single-router failure, single-arc failure, and single-router or single-arc failure.



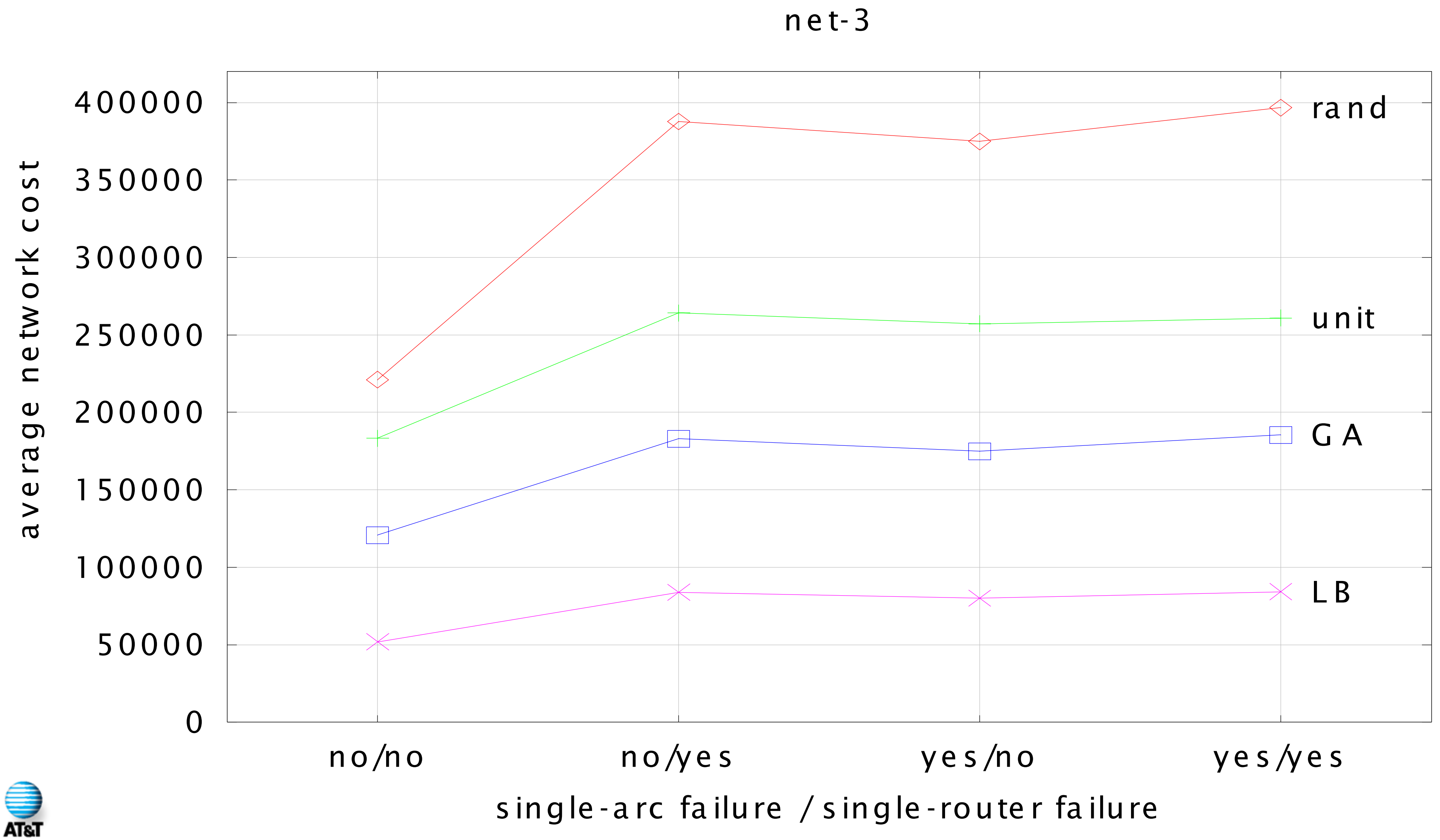
Average network costs for random weights, unit weights, GA weights compared to lower bound. Network has 10 routers, 90 arcs, 10 terminal nodes, and 90 demand pairs.



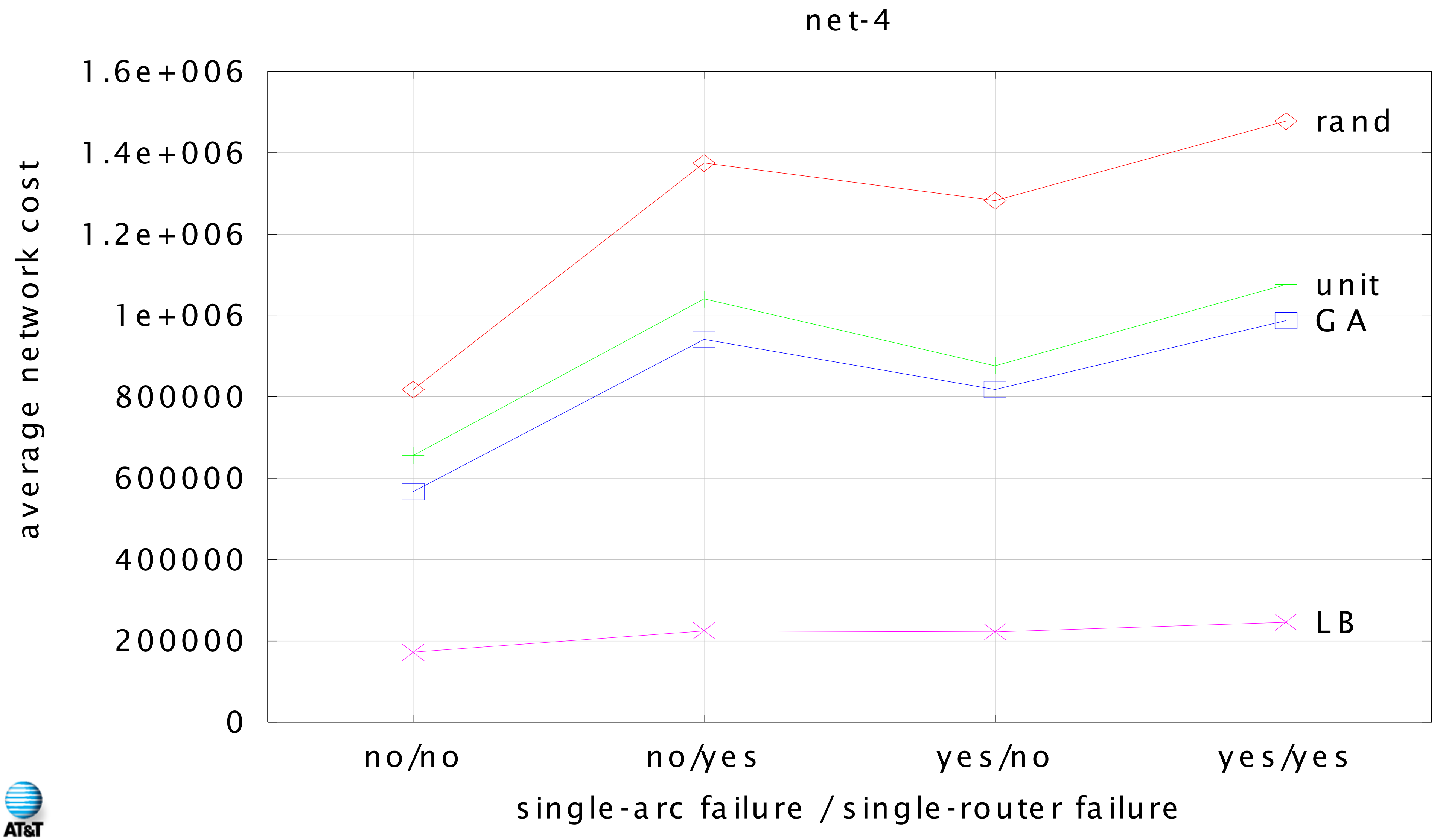
Average network costs for random weights, unit weights, GA weights compared to lower bound. Network has 11 routers, 110 arcs, 11 terminal nodes, and 110 demand pairs.



Average network costs for random weights, unit weights, GA weights compared to lower bound. Network has 74 routers, 278 arcs, 18 terminal nodes, and 306 demand pairs.



Average network costs for random weights, unit weights, GA weights compared to lower bound. Network has 71 routers, 350 arcs, 71 terminal nodes, and 4960 demand pairs.



Paper

L.S. Buriol, M.G.C. Resende, and M. Thorup, “Survivable IP network design with OSPF routing,” AT&T Labs Technical Report TD-64KUAW, September 2004.

Concluding Remarks

- we have seen a small sample of applications of optimization in telecommunications
- opportunities for optimization arise in practice all the time
- our profession call have a major impact in telecommunications

Concluding remarks

- These slides, and papers about GRASP, path-relinking, and their telecom applications available at:
  http://www.research.att.com/~mgcr
  http://graspheuristic.org

Handbook of Optimization in Telecommunications (HOT), P.M. Pardalos and M.G.C. Resende, eds. Springer, forthcoming in 2005.

- Dynamic programming
- Interior point methods for large-scale LP
- Decomposition methods in telecommunications
- Integer programming
- Lagrangean relaxation
- Minimum cost network flow algorithms
- Shortest path algorithms
- Multi-commodity flow in telecommunications
- Steiner tree problems in telecommunications
- Minimum spanning tree problems
- Metaheuristics
- Nonlinear programming
- Telecommunications network design
- Ring network design
- Computational large-scale linear programming
- Telecommunications access network design
- Network location in telecommunications
- Protection and dynamic optimization of optical networks
- Network location problems in telecommunications
- Optimization in wireless networks
- Optimization issues in combinatorial auctions
- New models for dynamic networks
- Optimization issues in distribution network design
- Optimization issues in network survivability
- Virtual path design
- Network grooming
- Network reliability in telecommunications
- Optimization issues in quality of service
- Frequency assignment problem
- Optimization in cellular phone networks
- Optimization issues in web search engines
- Optimization issues in IP routing
- Network planning in telecommunications
- Pricing and equilibrium in telecommunications
- Discrete multi-commodity network flow problems and applications in telecommunications
- Cliques and graph coloring in telecommunications
- Assignment problems
- Stochastic optimization in telecommunications
- Optimization issues in multicast trees
- Optimization of dynamic routing networks
- Stable paths in interdomain routing
- Network restoration
- Optimization in e-commerce
- Supernetworks

The End