Mauricio G. C. Resende

Algorithms & Optimization Research

AT&T Labs Research, Florham Park

Randomized heuristics for the MAX-CUT problem

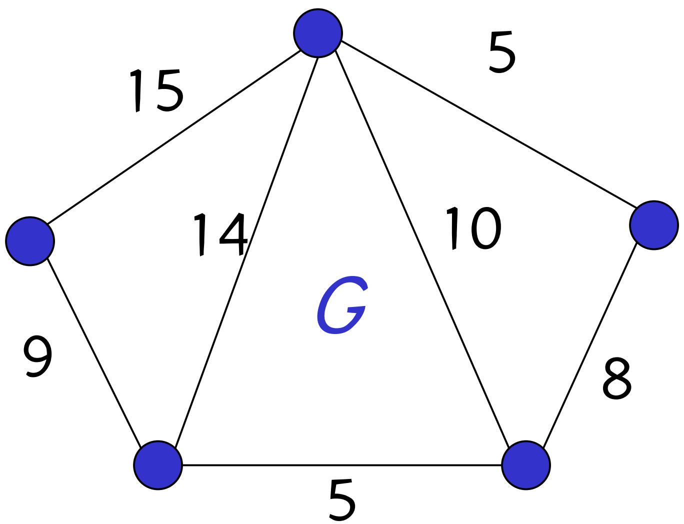
MAX-CUT

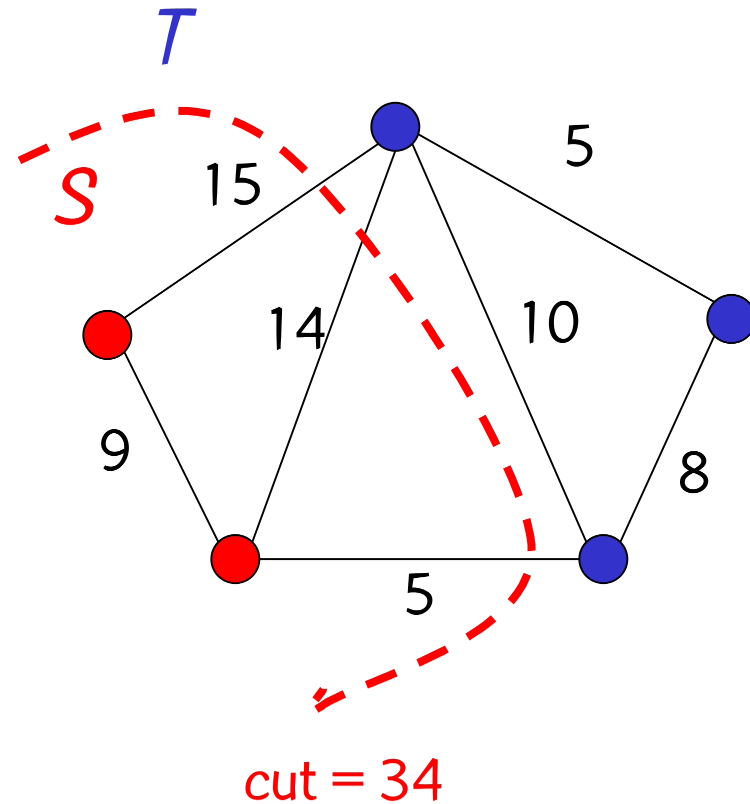
Given an undirected graph $G = (V, E)$ with weights weights $w_1, \ldots, w_m$ on the edges, find a vertex partition S, T such that the sum of the weights in the cut (S, T)

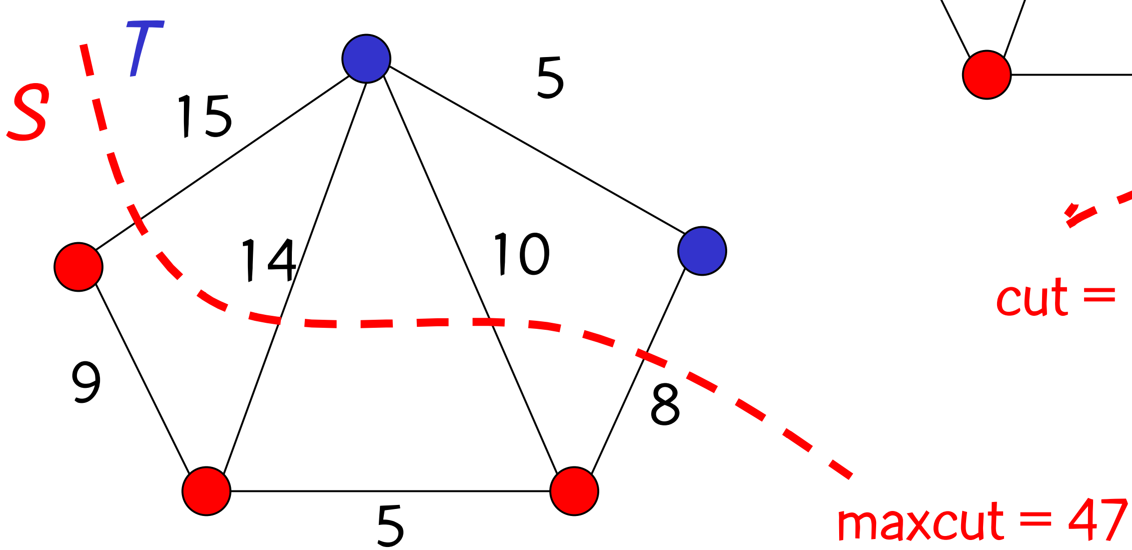
$$w(S,T) = \sum_{(i,j) \in E \ni i \in S, j \in T} w_{i,j}$$

is maximized.

MAX-CUT







MAX-CUT

- NP-hard (Karp, 1972) and remains NP-hard for unweighted version, i.e. with unit weights.
- Many applications, including:
  - VLSI design
  - Statistical physics
- .878-opt approximation algorithm of Goemans and Williamson (1995) based on semidefinite programming
- Success claimed by semidefinite programming research community on approximation algorithms for MAX-CUT.

Outline

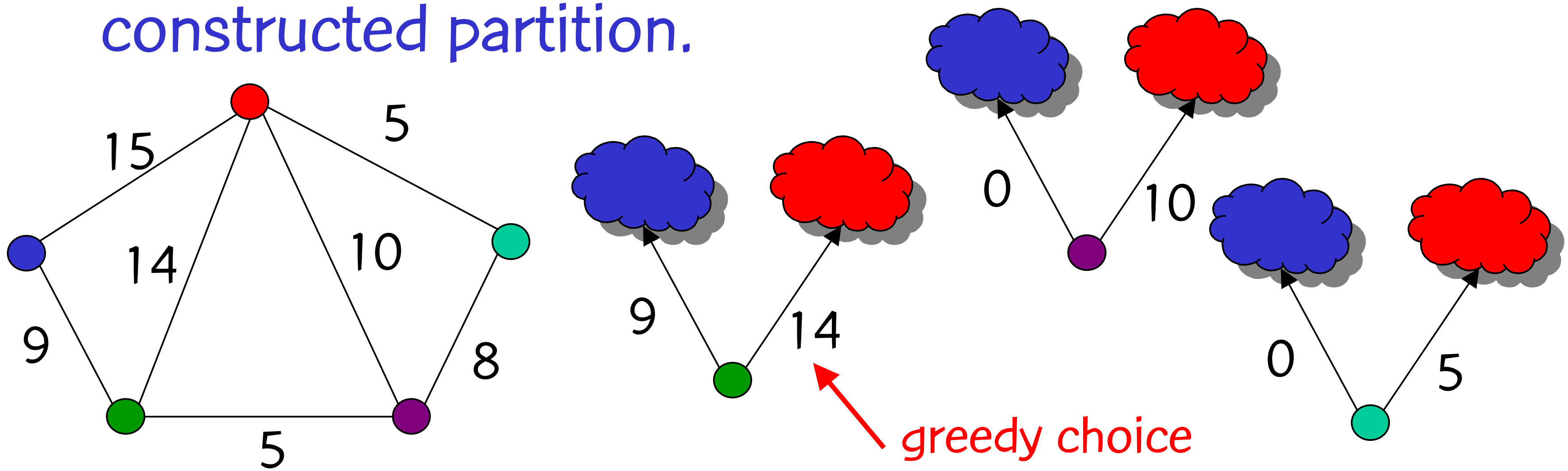
- Festa, Pardalos, R., and Ribeiro, “Randomized heuristics for the MAX-CUT problem,” Optimization Methods and Software, *in print.*
- GRASP
- Path-relinking (PR)
- Variable neighborhood search (VNS)
- Hybrids
- Computational study

GRASP for MAX-CUT

- Multi-start procedure (Feo & Resende, 1989) where each iteration consists of:
  - Construction of a greedy randomized feasible solution
  - Local search, starting from the constructed solution, produces a locally optimal solution

GRASP construction

- Initial edge is biased by its weight.
- Then, vertices are added, one at a time, biased by sum of weights of its edges incident to constructed partition.



Restricted candidate list (RCL) mechanism

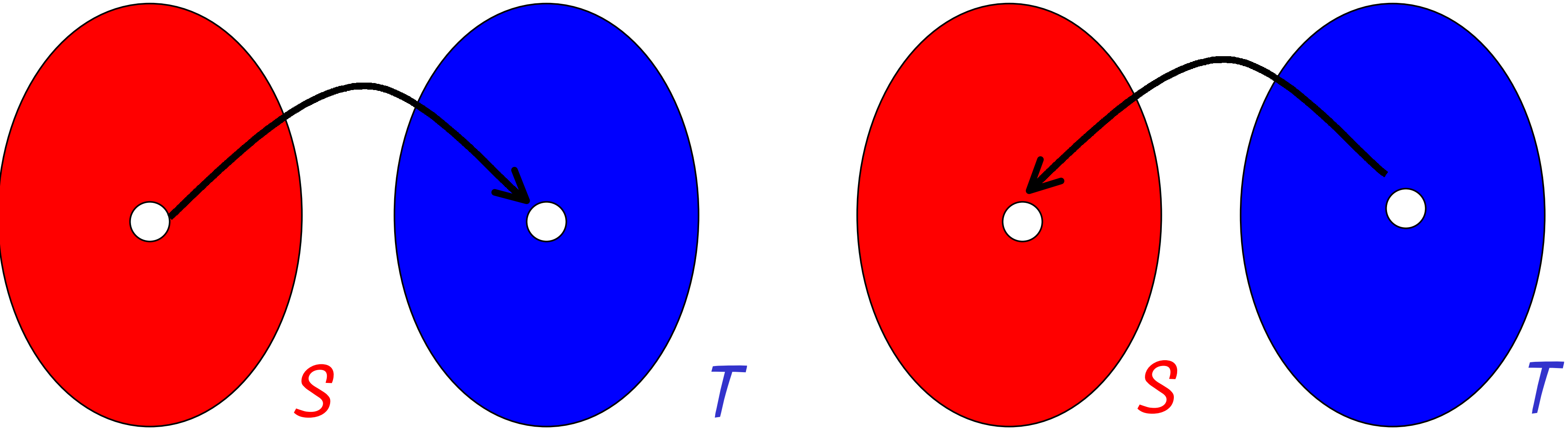
- Let $\sigma(v, S)$ and $\sigma(v, T)$ be the sum of edge weights between vertex v and partitions S and T, respectively.
- $\sigma^+ = \max \{\sigma(v, S), \sigma(v, T) \mid v \notin S \cup T\}$
- $\sigma^- = \min \{\sigma(v, S), \sigma(v, T) \mid v \notin S \cup T\}$
- RCL $= \{v \notin S \cup T \mid$

$$\sigma(v, S), \sigma(v, T) \geq \sigma^- + \alpha(\sigma^+ - \sigma^-)\}$$

$0 \leq \alpha \leq 1$

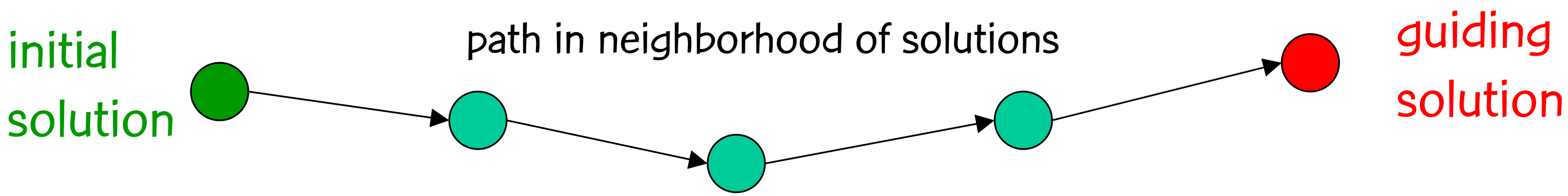
Local search

Neighborhood consists of all moves that change the partition of a single vertex.



Path-relinking (PR)

- Introduced in context of tabu search by Glover (1996):
  - Approach to integrate intensification & diversification in search.
- Consists in exploring trajectories that connect high quality solutions.



Path-relinking

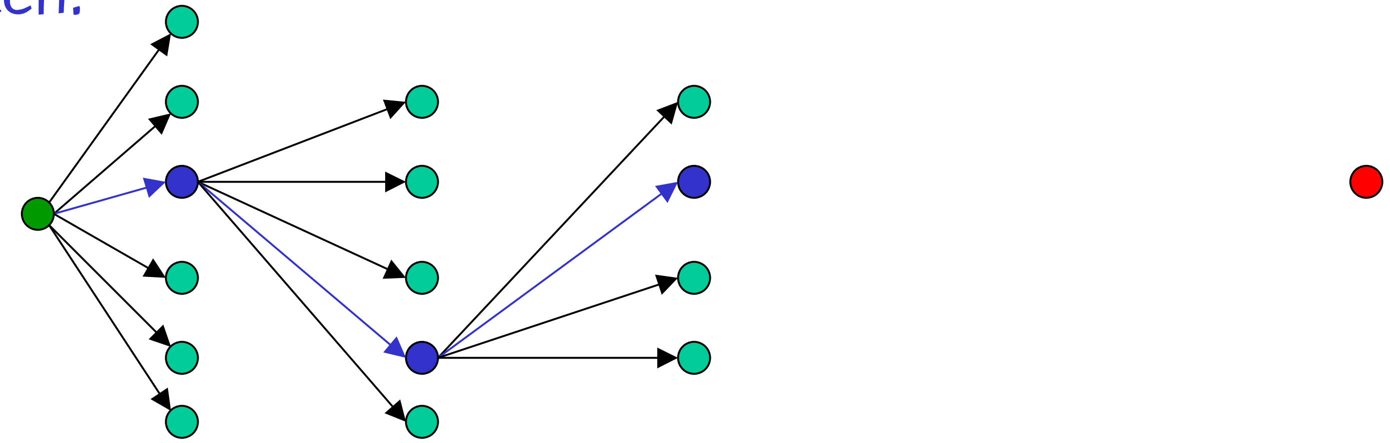
- Path is generated by selecting moves that introduce in the initial solution attributes of the guiding solution.
- At each step, all moves that incorporate attributes of the guiding solution are analyzed and best move is taken.

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 analyzed and best move is taken.

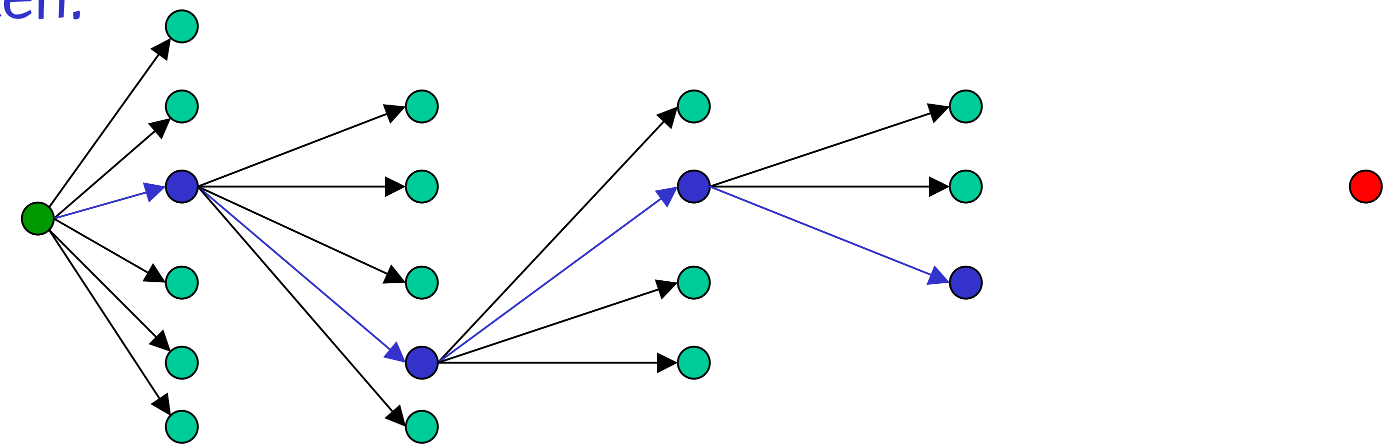
Path-relinking

- 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 analyzed and best move is taken.



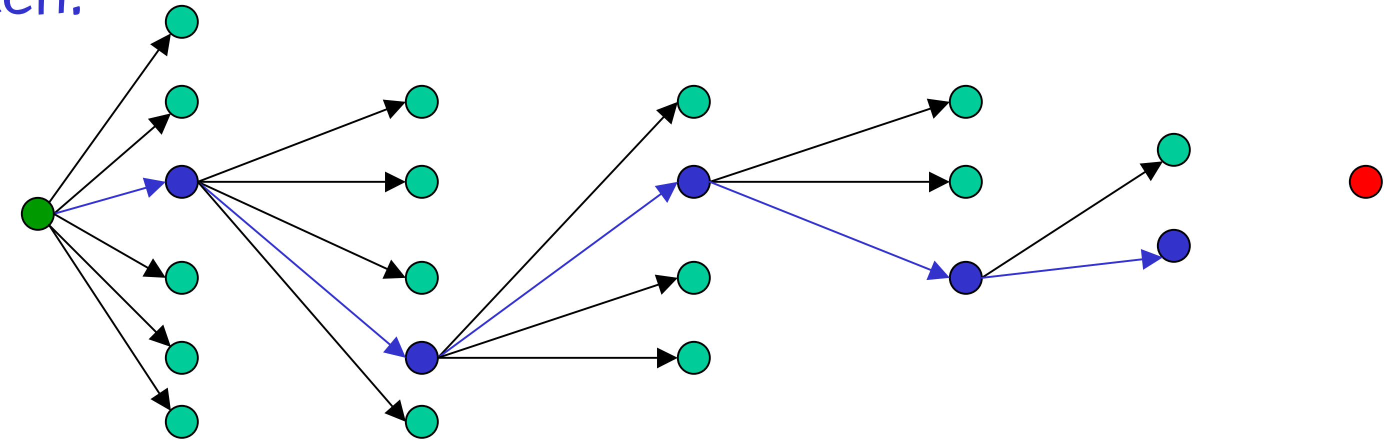
Path-relinking

- 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 analyzed and best move is taken.



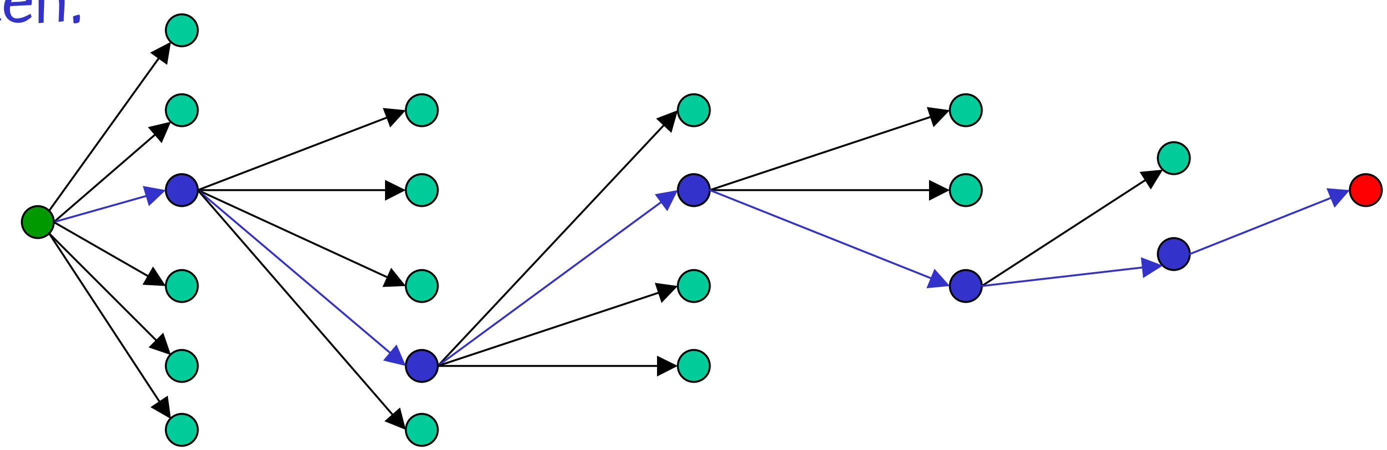
Path-relinking

- 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 analyzed and best move is taken.



Path-relinking

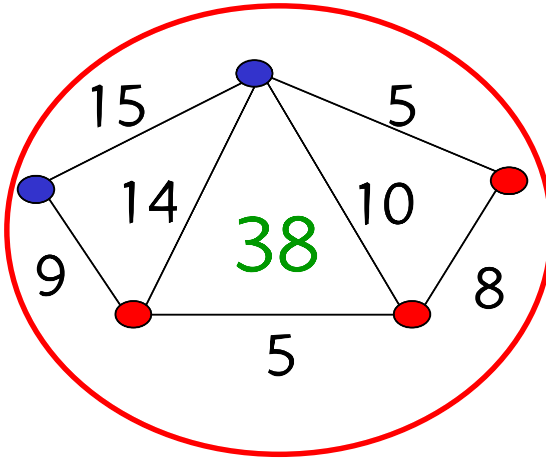
- Path is generated by selecting moves that introduce in the initial solution attributes of the guiding solution.
- At each step, all moves that incorporate attributes of the guiding solution are analyzed and best move is taken.



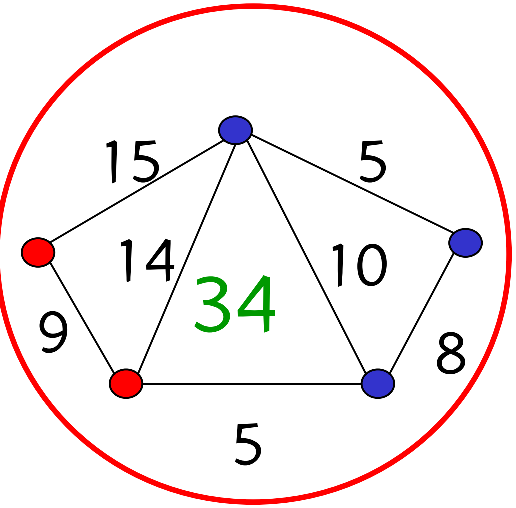
Path-relinking in GRASP

- Introduced by Laguna & Martí (1999)
- Maintain an elite set of solutions found during GRASP iterations.
- After each GRASP iteration (construction & local search):
  - Select an elite solution at random: guiding solution.
  - Use GRASP solution as initial solution.
  - Do path-relinking from initial solution to guiding solution.

Path-relinking for MAX-CUT

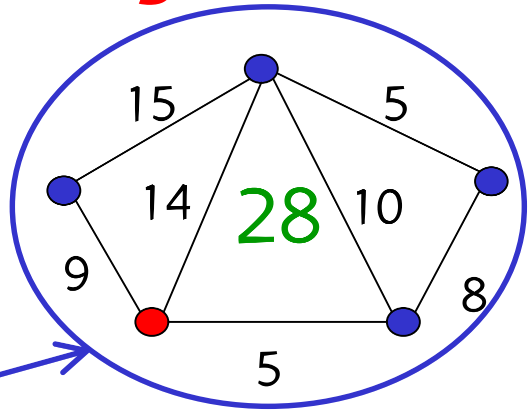


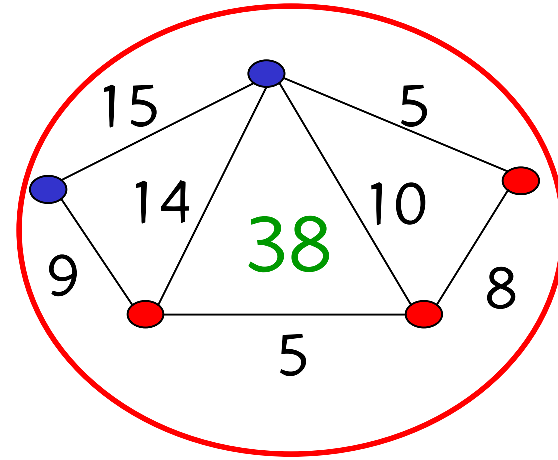
guiding solution



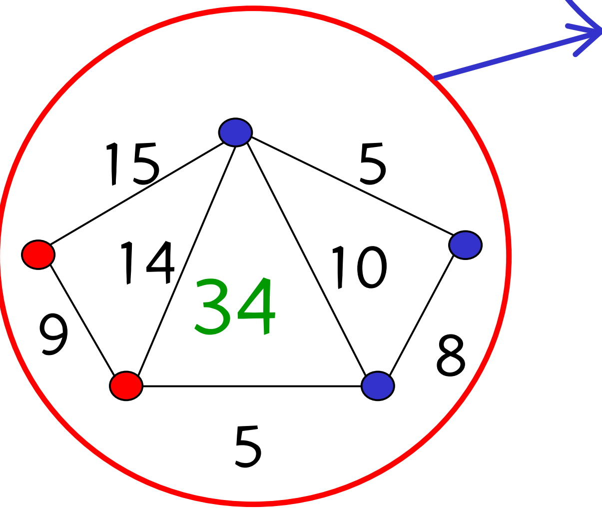
initial solution

Path-relinking for MAX-CUT





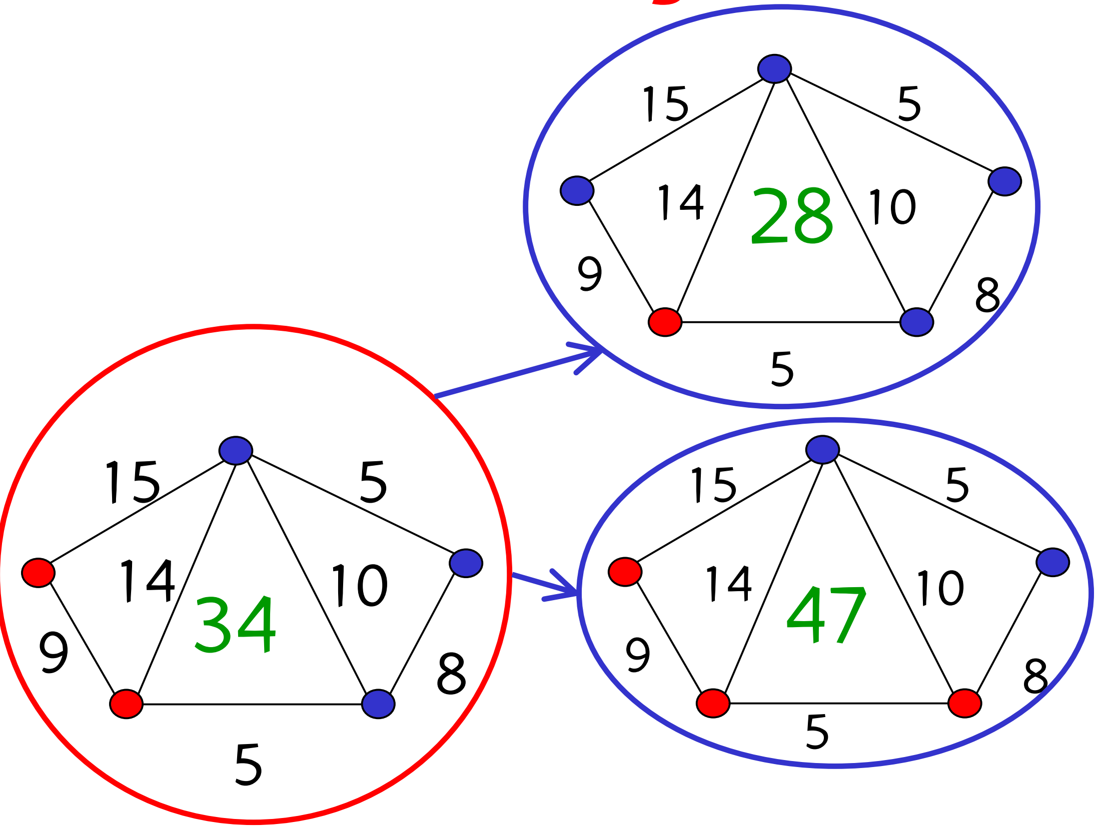
guiding solution



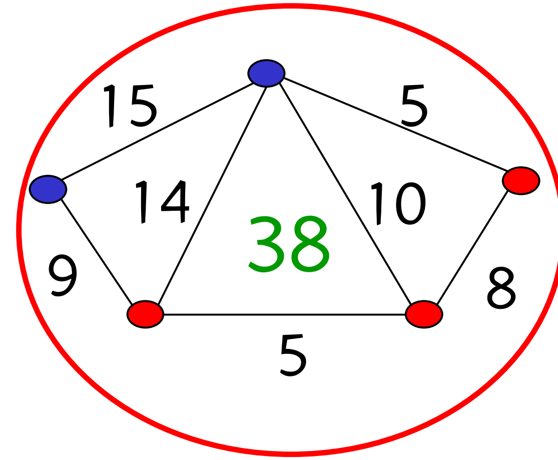
initial solution

5

Path-relinking for MAX-CUT



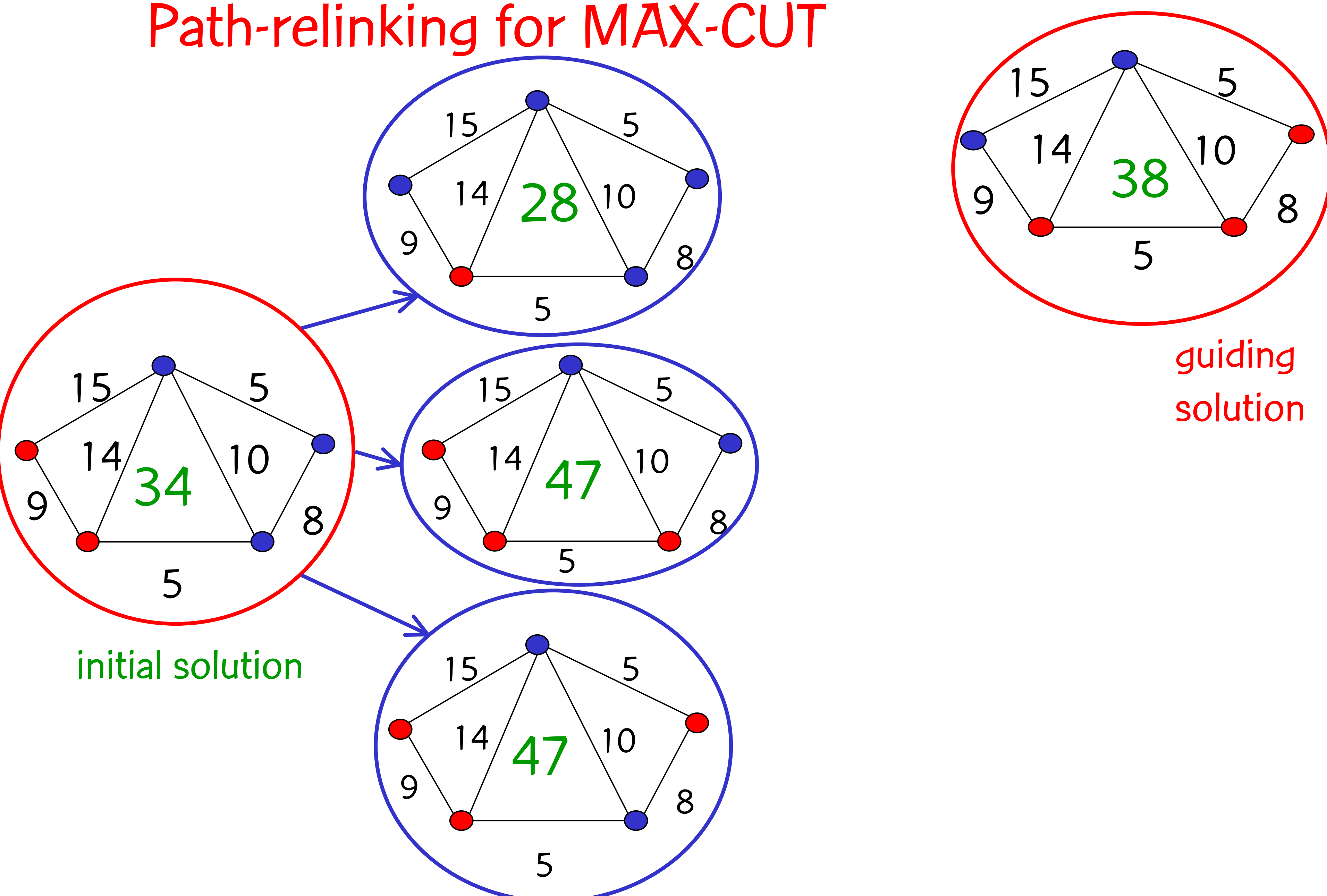
initial solution



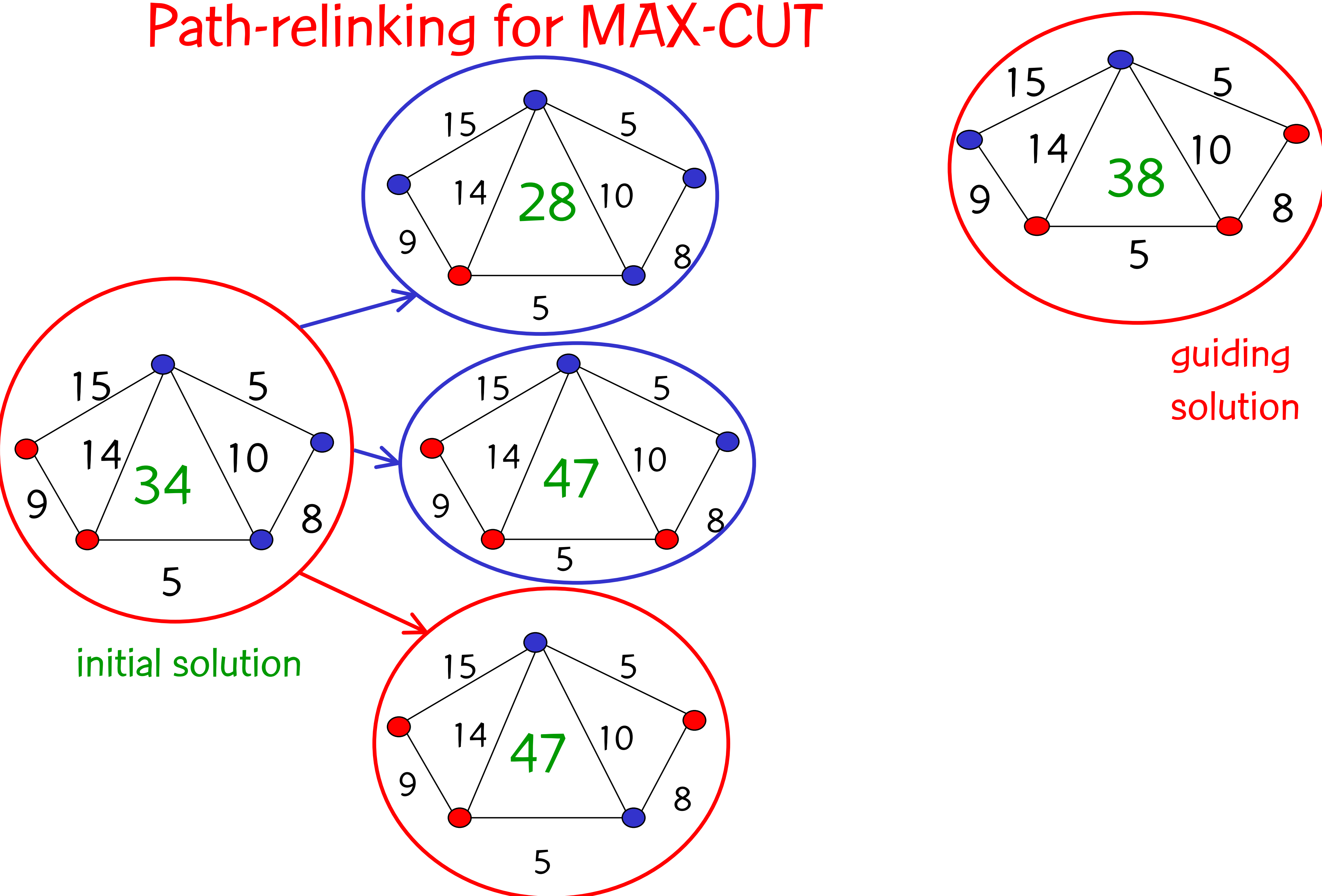
guiding solution

5

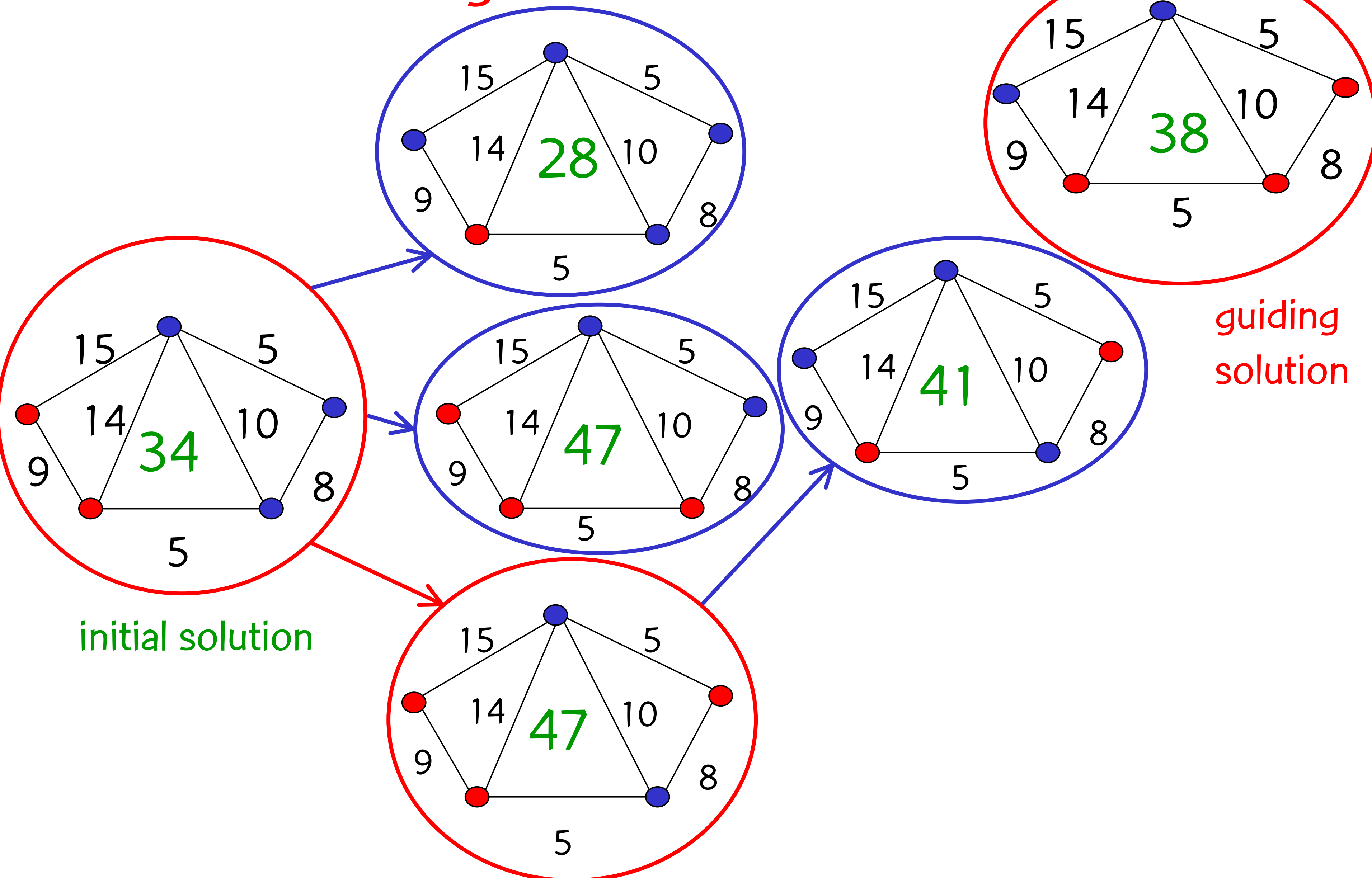
Path-relinking for MAX-CUT



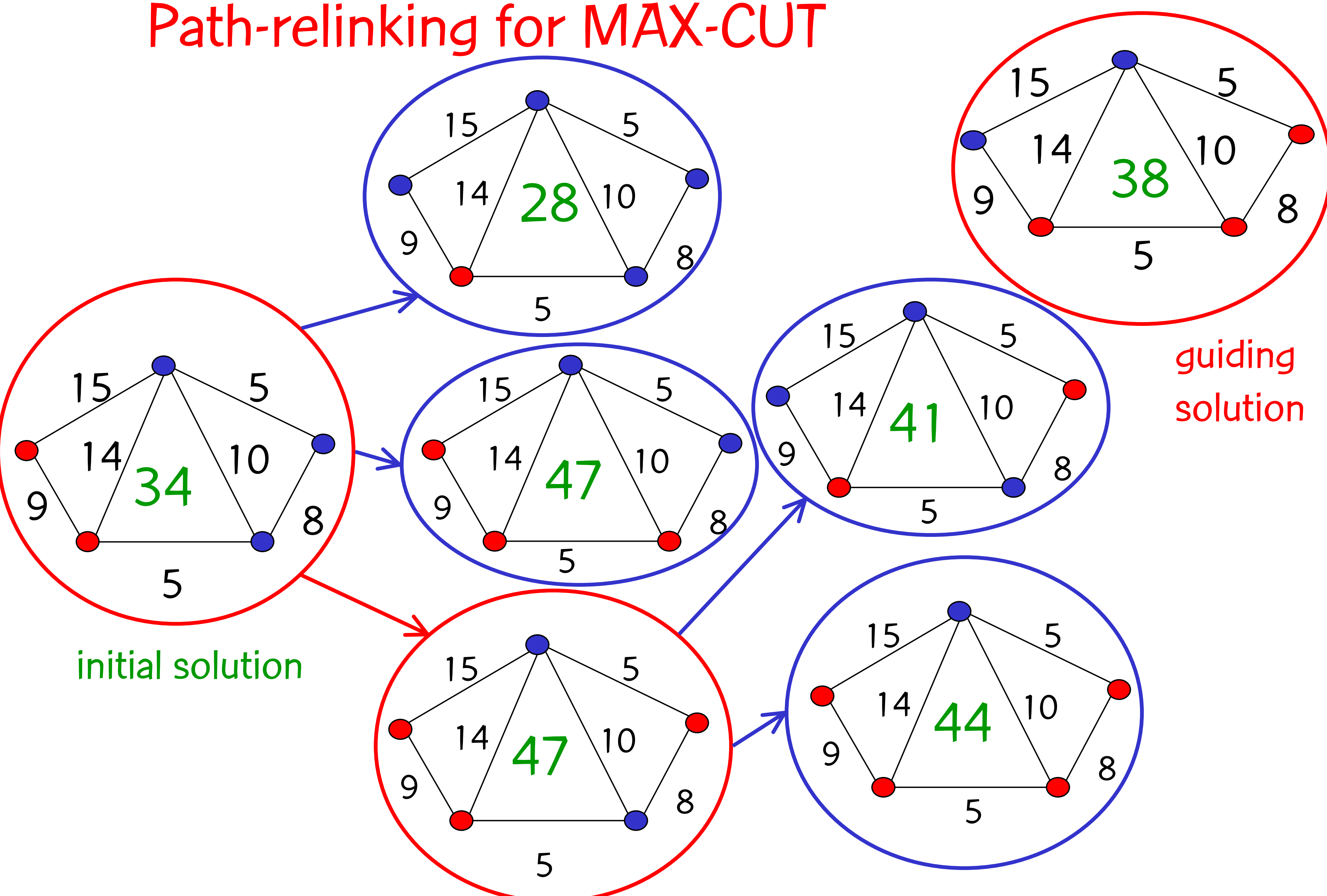
Path-relinking for MAX-CUT



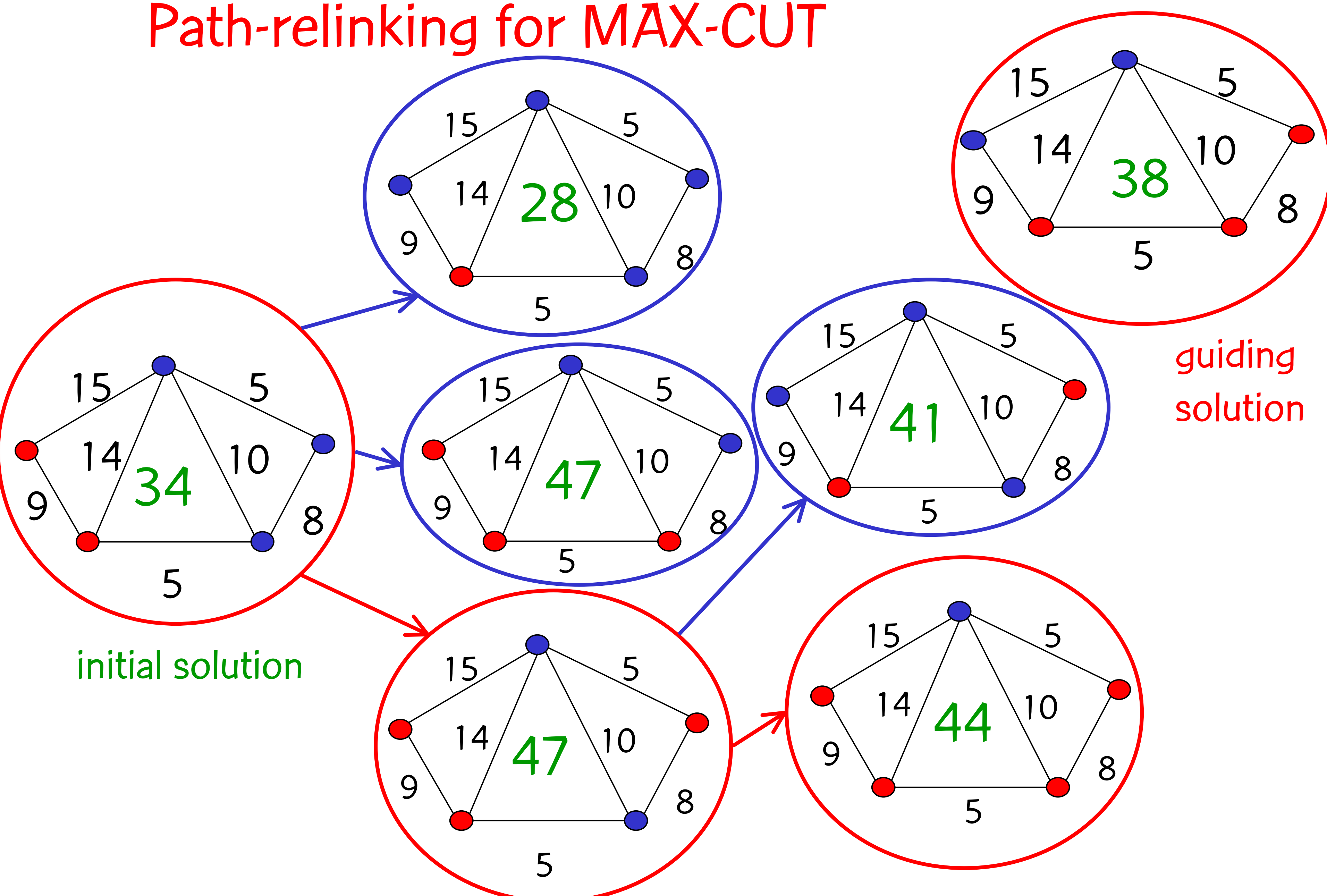
Path-relinking for MAX-CUT



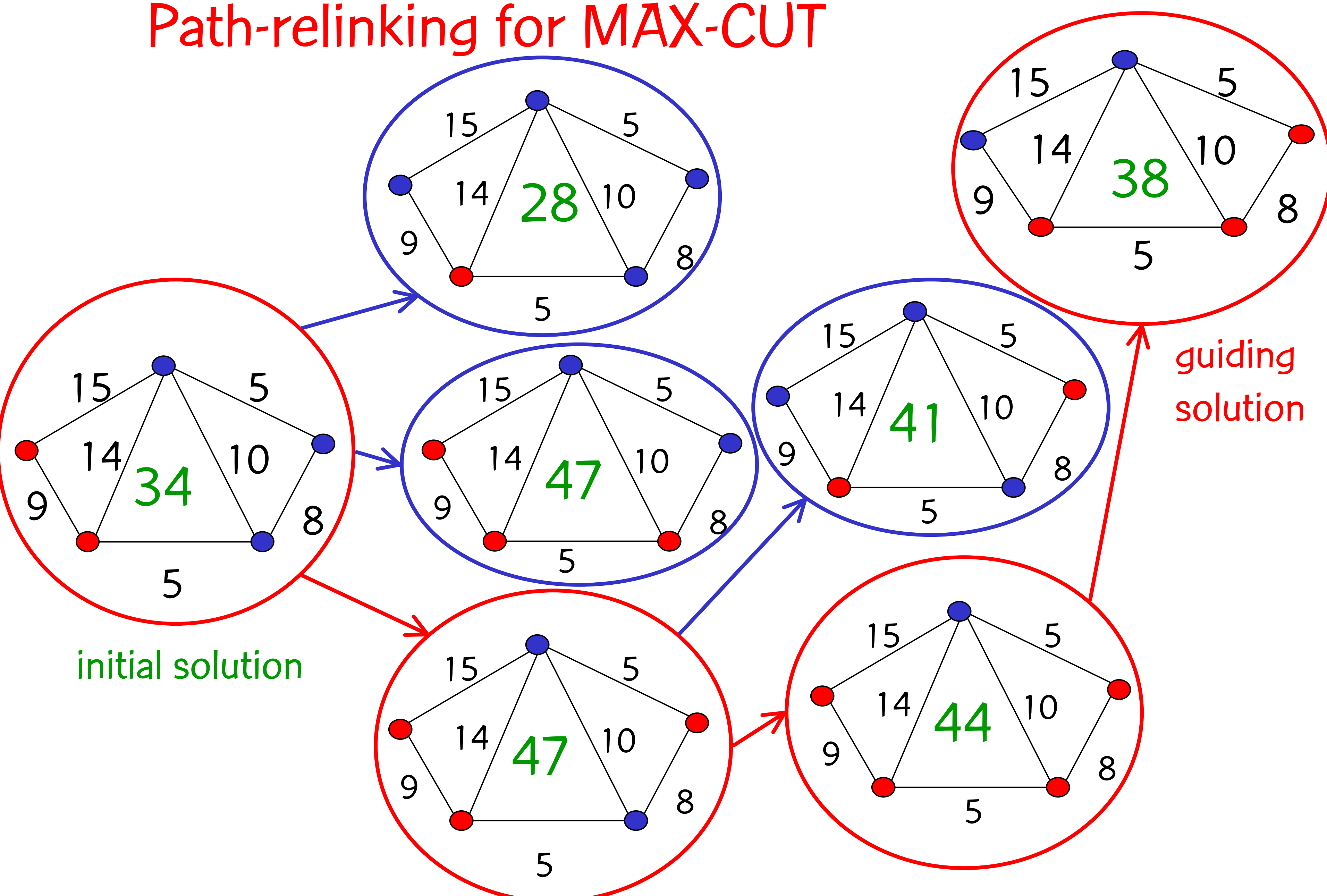
Path-relinking for MAX-CUT



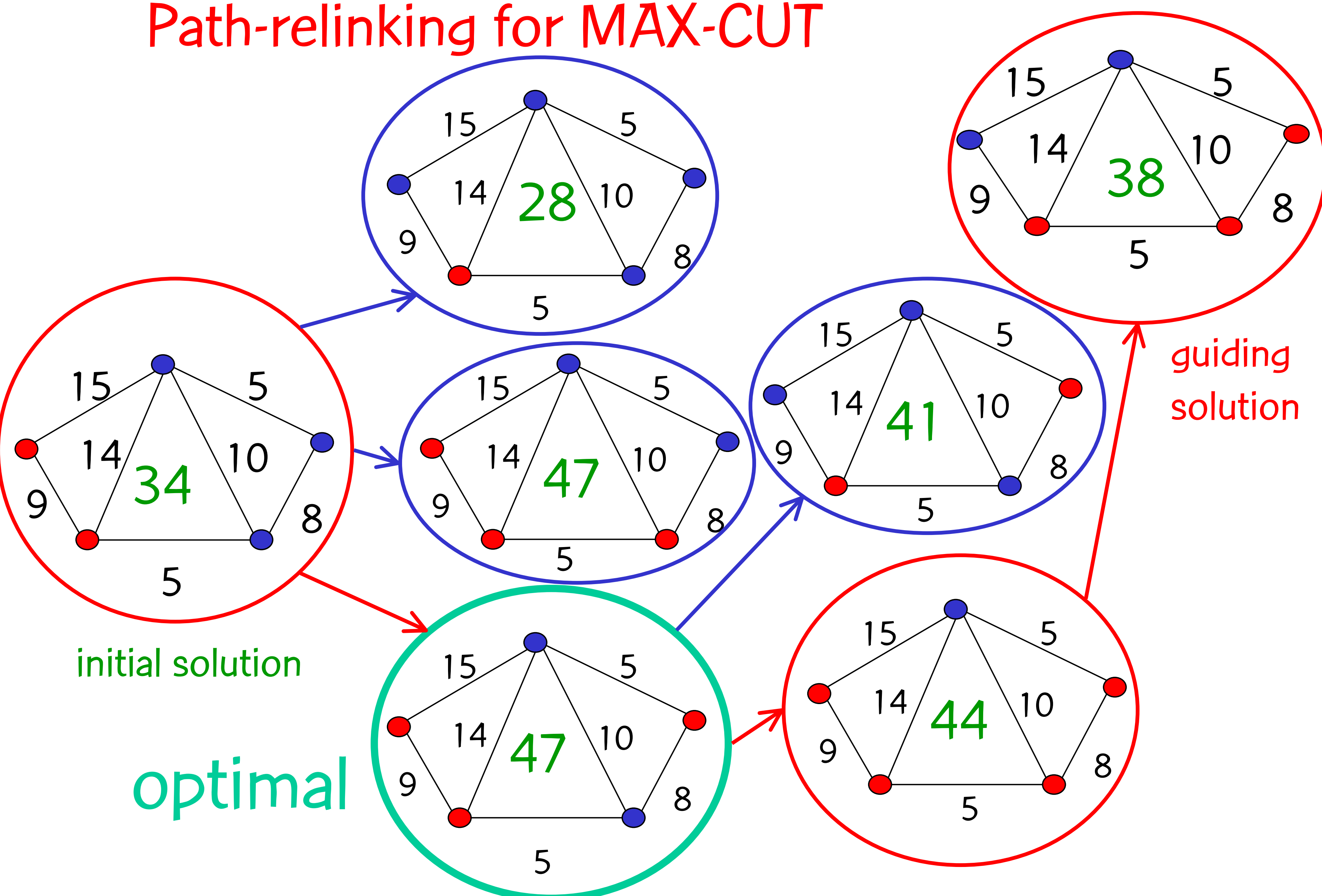
Path-relinking for MAX-CUT



Path-relinking for MAX-CUT



Path-relinking for MAX-CUT



VNS for MAX-CUT

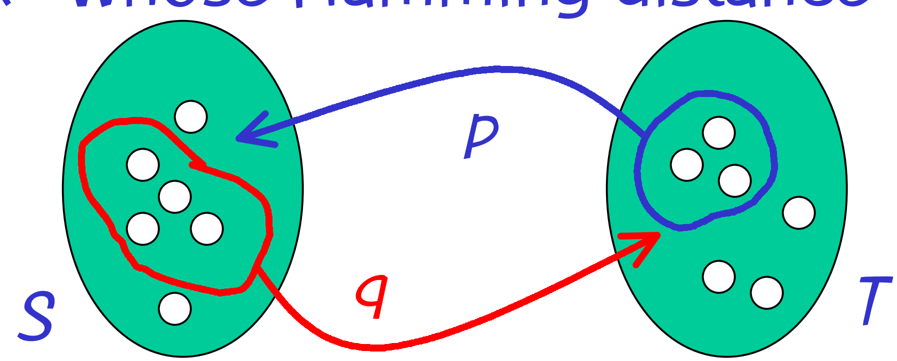
Variable Neighborhood Search (VNS):
Mladenović & Hansen (1997)

Let x represent a MAX-CUT solution, i.e.

$x_i = 1$ if $i \in S$,

$x_i = 0$ if $i \in T$.

The k-th order neighborhood $N^k(x)$ consists of all solutions x' whose Hamming distance from x is exactly k.



$p + q = k$

AT&T

VNS and GRASP + VNS for MAX-CUT

```
for t = 1,...,maxIterations {
    x = GenerateInitialSolution( );
    k = 1;
    for k ≤ k_max {
        randomly generate x' ∈ N^k(x);
        x'' = LocalSearch (x');
        if ( f(x'') > f(x) ){
            x = x'';
            k = 1;
        }
        else k = k+1;
    }
}
```

ConstructRandom( );

VNS

ConstructGreedyRandomized( );

GRASP + VNS

Randomized heuristics for MAX-CUT

- G : GRASP
- GPR : GRASP that uses path-relinking (PR) for intensification
- VNS : Variable neighborhood search (VNS)
- VNSPR : VNS that uses PR for intensification
- GVNS : GRASP that uses VNS in local search phase
- GVNSPR : GRASP that uses VNS in local search phase and PR for intensification

Computational experiments

- Single-iteration runs to compare to semidefinite programming upper bound & Choi & Ye's DSDP
- 1000-iteration runs to compare to Burer, Monteiro, and Zhang's CIRCUT
- Compare different variants:
  - GRASP
  - GRASP with path-relinking
  - VNS ($k_{max} = 100$)
  - VNS ($k_{max} = 100$) with path-relinking
  - GRASP with VNS ($k_{max} = 15$)
  - GRASP with VNS ($k_{max} = 15$) with path-relinking

All runs on SGI Challenge computer (196MHz R10000 processor).
All algorithms coded in f77 or f90.

Test problems

- Used by semidefinite programming research community
- Type I: Generated by Helmberg & Rendl (1997) with a network generator written by G. Rinaldi
- Type II: Spin glass instances from the 7th DIMACS Implementation Challenge by Jünger & Liers.
- Type III: Instances on cubic lattice graphs, modeling Ising spin glasses, proposed by Burer et al. (2001).

Single-iteration randomized algorithms X interior-point SDP code

Cut value			single iteration		
PROB	dim	DSDP	G	GVNS	VNS
G14	800 x 4694	2922	3009	3011	3040
G15	800 x 4661	2938	2978	3008	3017
G22	2000 x 19990	12960	13027	13156	13087
G23	2000 x 19990	13006	13121	13181	13190
G24	2000 x 19990	12933	13059	13097	13209
G50	3000 x 6000	5880	5812	5838	5820

DSDP (Choi & Ye, 2000): A fast dual interior point code for SDP.

Red cell indicates best solution.

Single-iteration randomized algorithms X interior-point SDP code

Time (SGI seconds)

PROB	dim	DSDP	G (single iteration)	GVNS (single iteration)	VNS (single iteration)
G14	800 x 4694	17	0.5	1.8	13
G15	800 x 4661	15	0.5	3.6	18
G22	2000 x 19990	1982	6.3	43	57
G23	2000 x 19990	1555	6.3	42	141
G24	2000 x 19990	1563	6.6	46	193
G50	3000 x 6000	127	3.9	19	76

DSDP (Choi & Ye, 2000): A fast dual interior point code for SDP.

Red cell indicates best solution time.

Single-iteration randomized algorithms without path-relinking X 1000-iteration randomized algorithms with path-relinking

Cut value

			single iteration			1000 iterations		
PROB	dim	[redacted]	G	GVNS	VNS	GPR	GVNSPR	VNSPR
G14	800 x 4694		3009	3011	3040	3041	3044	3055
G15	800 x 4661		2978	3008	3017	3034	3031	3043
G22	2000 x 19990		13027	13156	13087	13203	13246	13295
G23	2000 x 19990		13121	13181	13190	13222	13260	13290
G24	2000 x 19990		13059	13097	13209	13242	13255	13276
G50	3000 x 6000		5812	5838	5820	5880	5880	5880

Red cell indicates best solution.

1000-iteration randomized algorithms X interior-point SDP code

Cut value		[redacted]		[redacted]	1000 iterations		
PROB	dim	[redacted]	DSDP	[redacted]	GPR	GVNSPR	VNSPR
G14	800 x 4694	[redacted]	2922	[redacted]	3041	3044	3055
G15	800 x 4661	[redacted]	2938	[redacted]	3034	3031	3043
G22	2000 x 19990	[redacted]	12960	[redacted]	13203	13246	13295
G23	2000 x 19990	[redacted]	13006	[redacted]	13222	13260	13290
G24	2000 x 19990	[redacted]	12933	[redacted]	13242	13255	13276
G50	3000 x 6000	[redacted]	5880	[redacted]	5880	5880	5880

DSDP (Choi & Ye, 2000): A fast dual interior point code for SDP.

Red cell indicates best solution.

AT&T

1000-iteration randomized algorithms X interior-point SDP code

Time (SGI seconds)

			1000 iterations		
PROB	dim	DSDP	GPR	GVNSPR	VNSPR
G14	800 x 4694	17	489	2337	16734
G15	800 x 4661	15	488	2495	17184
G22	2000 x 19990	1982	6724	32175	197654
G23	2000 x 19990	1555	6749	31065	193707
G24	2000 x 19990	1563	6697	31143	195749
G50	3000 x 6000	127	5095	16217	147132

DSDP (Choi & Ye, 2000): A fast dual interior point code for SDP.

Red cell indicates best solution time.

1000-iteration randomized algorithms X interior-point SDP code

Cut value			[redacted]	1000 iterations		
PROB	dim	CIRCUT		GPR	GVNSPR	VNSPR
G14	800 x 4694	3053		3041	3044	3055
G15	800 x 4661	3039		3034	3031	3043
G22	2000 x 19990	13331		13203	13246	13295
G23	2000 x 19990	13269		13222	13260	13290
G24	2000 x 19990	13287		13242	13255	13276
G50	3000 x 6000	5856		5880	5880	5880

CIRCUT (Burer, Monteiro, & Zhang, 2001): rank-2 relaxation heuristic.

Red cell indicates best solution.

1000-iteration randomized algorithms X interior-point SDP code

Time (SGI seconds)

PROB	dim	CIRCUT	[redacted]	1000 iterations GPR	GVNSPR	VNSPR
G14	800 x 4694	128		489	2337	16734
G15	800 x 4661	155		488	2495	17184
G22	2000 x 19990	493		6724	32175	197654
G23	2000 x 19990	457		6749	31065	193707
G24	2000 x 19990	521		6697	31143	195749
G50	3000 x 6000	231		5095	16217	147132

CIRCUT (Burer, Monteiro, & Zhang, 2001): rank-2 relaxation heuristic.

Red cell indicates best solution time.

Remarks

- DSDP is not competitive with randomized heuristics (nor with CIRCUT).
- VNS with path-relinking produces the best solutions amongst randomized heuristics.
- CIRCUT is fastest and produces good-quality solutions.

Ratio of cut found and SDP upper bound on single iteration randomized heuristics.

		GRASP		GVNS		VNS	
name	size (n, den)	cut/bnd	time	cut/bnd	time	cut/bnd	time
G1	800, 6.12%	.95	2s	.95	6s	.96	41s
G2		.95	2s	.95	3s	.96	37s
G3		.95	2s	.95	5s	.96	17s
G14	800, 1.58%	.94	.5s	.94	2s	.95	13s
G15		.94	.5s	.95	4s	.95	18s
G16		.94	.5s	.94	2s	.95	10s

Time on SGI Challenge (196MHz R10000 processor)

Red cell indicates best solution.

Ratio of cut found and SDP upper bound on single iteration randomized heuristics.

		GRASP		GVNS		VNS	
name	size (n, den)	cut/bnd	time	cut/bnd	time	cut/bnd	time
G22	2000, 1.05%	.92	6s	.93	43s	.93	57s
G23		.93	6s	.93	42s	.93	141s
G24		.92	6s	.93	46s	.93	193s
G35	2000, 0.64%	.94	4s	.95	17s	.95	143s
G36		.94	4s	.94	22s	.95	186s
G35		.94	4s	.95	17s	.95	205s

Time on SGI Challenge (196MHz R10000 processor)

Red cell indicates best solution.

Ratio of cut found and SDP upper bound on single iteration randomized heuristics.

		GRASP		GVNS		VNS	
name	size (n, den)	cut/bnd	time	cut/bnd	time	cut/bnd	time
G43	1000, 2.10%	.93	1s	.94	6s	.94	36s
G44		.93	1s	.93	5s	.93	41s
G45		.93	1s	.93	7s	.93	24s
G48	3000, 0.17%	.98	4s	1.0	11s	1.0	50s
G49		.99	2s	.99	8s	.98	52s
G50		.97	4s	.97	19s	.97	75s

Time on SGI Challenge (196MHz R10000 processor)

Red cell indicates best solution.

Ratio of cut found and SDP upper bound for CIRCUT and 1000-iteration randomized heuristics.

Name	CIRCUT	GRASP	+PR	GVNS	+PR	VNS	+PR
G1	**.9624**	.9555	.9573	.9574	.9595	.9622	.9622
G2	**.9614**	.9572	.9572	.9598	.9598	.9612	.9612
G3	**.9623**	.9564	.9593	.9602	.9602	**.9623**	**.9623**
G11	.8931	.8804	**.8995**	.8931	**.8995**	.8931	**.8995**
G12	.8889	.8792	.8889	.8857	**.8953**	.8921	**.8953**
G13	.8899	.8868	**.8992**	.8930	.8961	**.8992**	**.8992**

Red cell indicates best solution.

Ratio of cut found and SDP upper bound for CIRCUT and 1000-iteration randomized heuristics.

Name	CIRCUT	GRASP	+PR	GVNS	+PR	VNS	+PR
G14	**.9595**	.9498	.9542	.9551	.9551	.9586	.9586
G15	**.9621**	.9508	.9574	.9565	.9565	.9602	.9602
G16	**.9600**	.9499	.9546	.9555	.9555	.9593	.9593
G22	**.9450**	.9336	.9349	.9379	.9379	.9414	.9414
G23	.9425	.9345	**.9458**	.9384	.9385	.9406	.9406
G24	**.9422**	.9316	.9371	.9380	.9380	.9395	.9395

Red cell indicates best solution.

Ratio of cut found and SDP upper bound for CIRCUT and 1000-iteration randomized heuristics.

Name	CIRCUT	GRASP	+PR	GVNS	+PR	VNS	+PR
G32	.8910	.8782	.8923	.8859	.8936	.8885	**.8949**
G33	.8848	.8770	.8861	.8822	.8900	.8861	**.8953**
G34	.8877	.8748	.8851	.8825	.8877	.8877	**.8903**
G35	**.9587**	.9459	.9485	.9506	.9506	.9544	.9544
G36	**.9580**	.9448	.9481	.9510	.9510	.9545	.9545
G37	**.9572**	.9459	.9492	.9491	.9499	.9543	.9543

Red cell indicates best solution.

Ratio of cut found and SDP upper bound for CIRCUT and 1000-iteration randomized heuristics.

Name	CIRCUT	GRASP	+PR	GVNS	+PR	VNS	+PR
G43	.9472	.9381	.9422	.9424	.9424	**.9476**	**.9476**
G44	**.9460**	.9381	.9425	.9447	.9447	.9459	.9459
G45	**.9476**	.9399	.9430	.9443	.9443	.9467	.9467
G48	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
G49	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
G50	**.9820**	**.9790**	**.9820**	**.9776**	**.9820**	**.9800**	**.9820**

Red cell indicates best solution.

Remarks

- All heuristics were, on average, between 4.5% and 5.4% of the SDP upper bound;
- For randomized heuristics G, GVNS, and VNS the incorporation of path-relinking was beneficial;
- At the expense of longer running times, the use of VNS in the local search phase of GRASP was beneficial.
- At the expense of even longer running times, using larger neighborhoods in the pure VNS was beneficial.

Remarks

- Among the randomized heuristics, VNSPR found the best cuts;
- CIRCUT found slightly better solutions than VNSPR on 13 of the 24 instances, found slightly worse solutions on 7 of 24, and cuts of the same size were found on the remaining 4 instances;
- Overall, solutions found by CIRCUT and VNSPR differed by less than 0.12%.

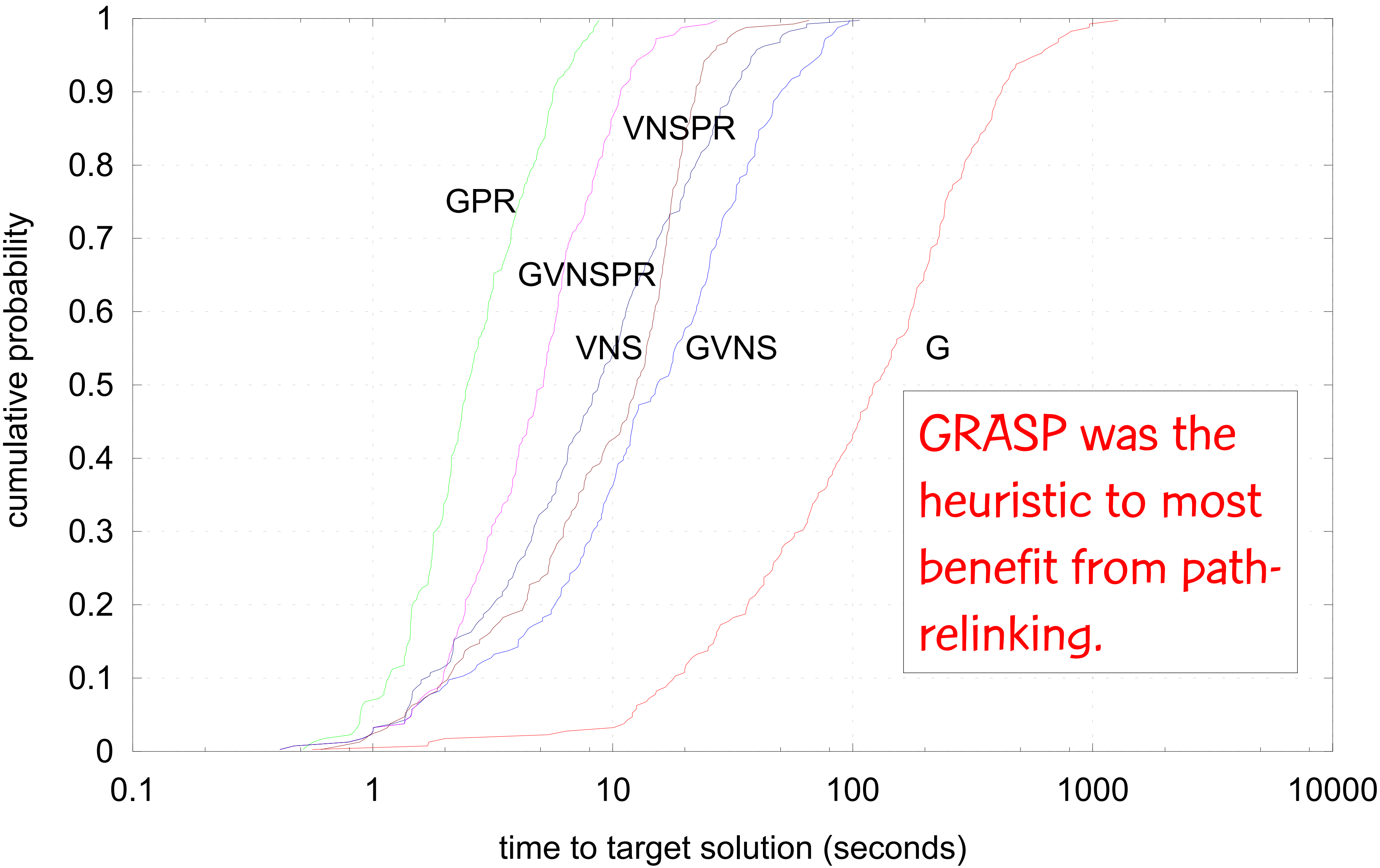
Remarks

- CIRCUT tended to find better solutions on densest problems while VNSPR did so on sparsest problems;
- Running times for 1000 iterations of the randomized heuristics went from a factor of 11 w.r.t. CIRCUT to over a factor of 300.

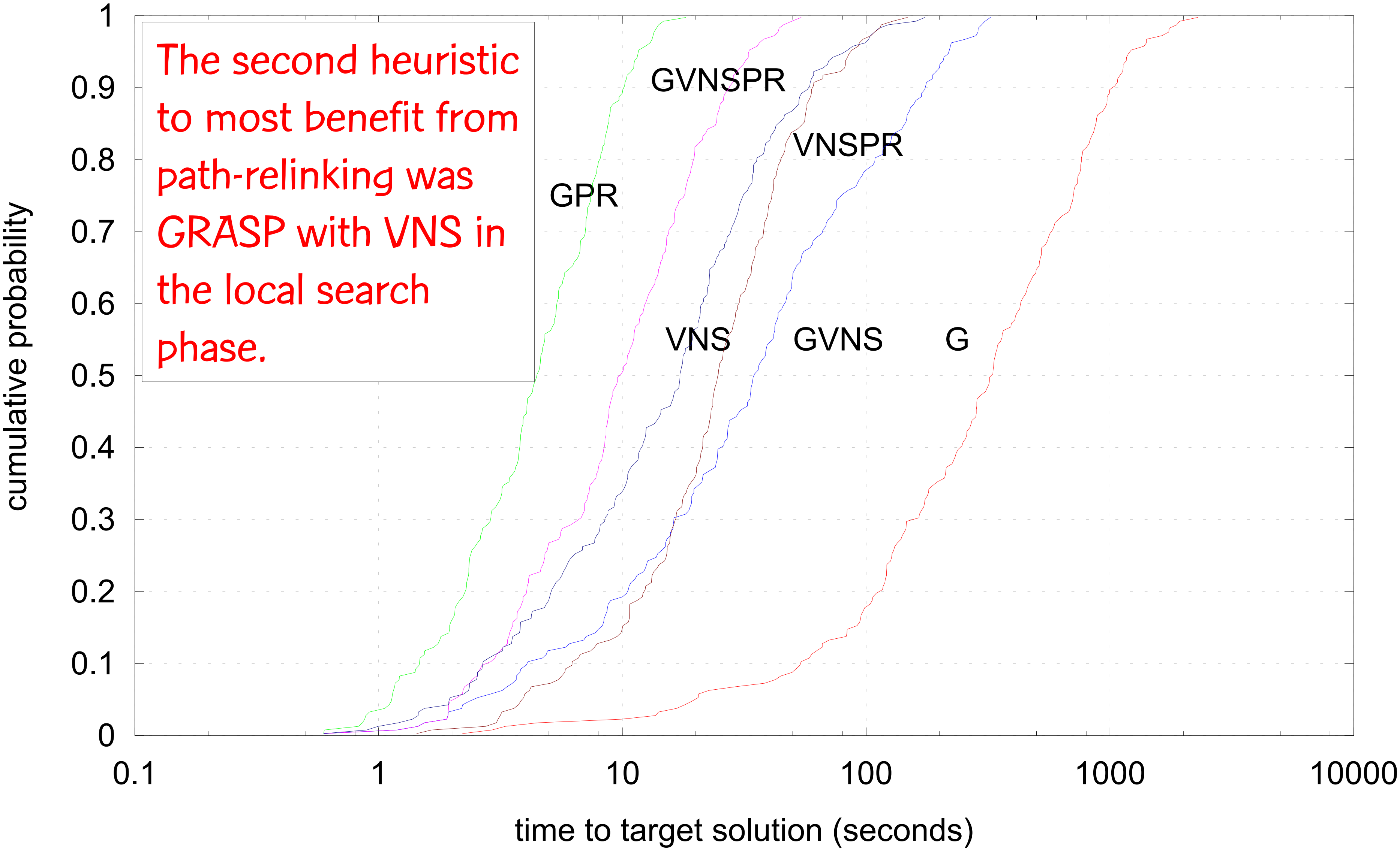
Solution time distribution

- Since running times of randomized heuristics vary substantially, we study their empirical distributions of the random variable *time-to-target-solution-value* considering G11, G12, and G13.
- Target values are values found by GRASP in the 1000 iteration runs, i.e. 552, 546, and 572, respectively.
- 200 independent runs of each heuristic were performed and running times to find target solutions recorded.

G11 (target solution: 552)

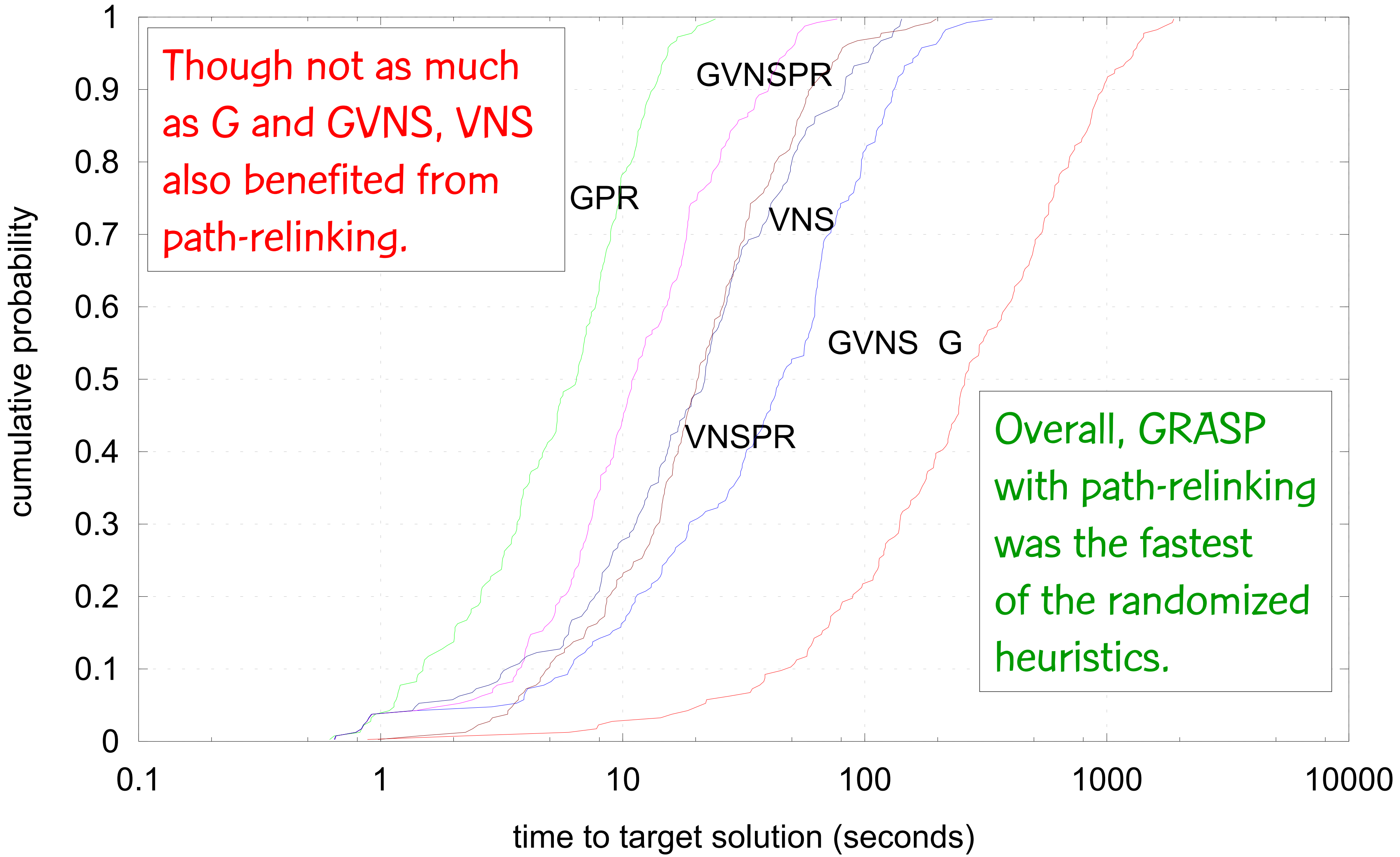


G12 (target solution: 546)



The second heuristic to most benefit from path-relinking was GRASP with VNS in the local search phase.

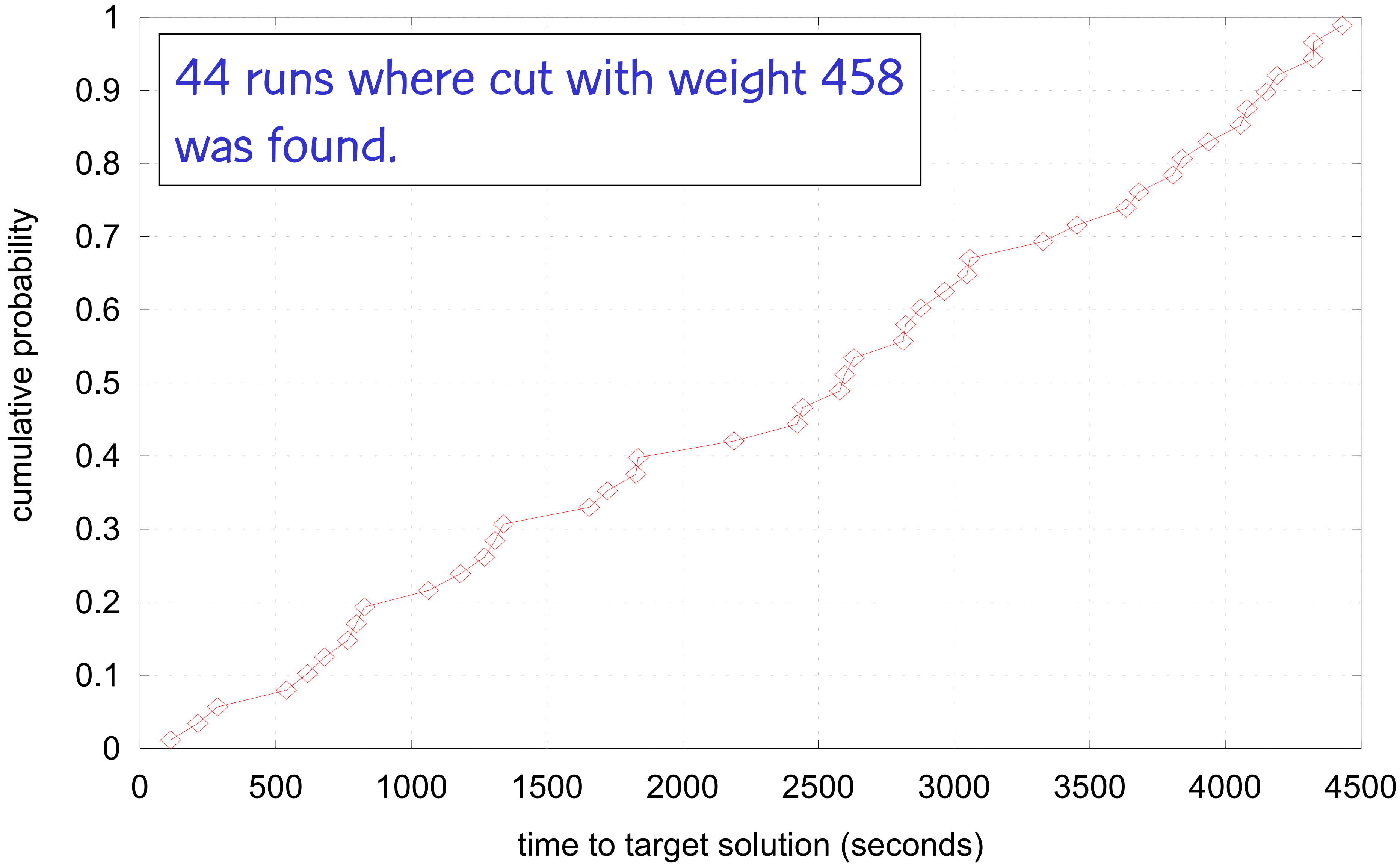
G13 (target solution: 572)



pm3-8-50: 7th DIMACS Challenge instance

- 512 nodes, 1.17% density, generated by Jünger & Liers using Ising model of spin glasses.
- Best known cut was 456 and best known upper bound is 461. Burer et al. (2001) report finding 454 with CIRCUT.
- We ran VNSPR 60 times for 1000 iterations each:
  - In 16 runs, VNSPR found cut of weight 456;
  - On the remaining 44, a new least-weight cut of value 458 was found.

pm3-8-50 target value = 458



1000-node (density = 0.60%)

Instances on cubic lattice graphs, modeling Ising spin glasses, proposed by Burer et al. (2001).

Cut value

problem	CIRCUT	GPR
S1	880	884
S2	892	896
S3	882	878
S4	894	884
S5	882	868
S6	886	870
S7	894	890
S8	847	876
S9	890	884
S10	886	888

1000-node (density = 0.60%)

Instances on cubic lattice graphs, modeling Ising spin glasses, proposed by Burer et al. (2001).

Cut value

problem	CIRCUT		GVNSPR
S1	880		884
S2	892		896
S3	882		878
S4	894		890
S5	882		874
S6	886		880
S7	894		892
S8	847		878
S9	890		896
S10	886		886

1000-node (density = 0.60%)
Instances on cubic lattice graphs,
modeling Ising spin glasses,
proposed by Burer et al. (2001).

Cut value

problem	CIRCUT	VNSPR
S1	880	892
S2	892	900
S3	882	884
S4	894	896
S5	882	882
S6	886	880
S7	894	896
S8	847	880
S9	890	898
S10	886	890

1000-node (density = 0.60%)
Instances on cubic lattice graphs, modeling Ising spin glasses, proposed by Burer et al. (2001).

Cut value

problem	CIRCUT	GPR	GVNSPR	VNSPR
S1	880	884	884	892
S2	892	896	896	900
S3	882	878	878	884
S4	894	884	890	896
S5	882	868	874	882
S6	886	870	880	880
S7	894	890	892	896
S8	847	876	878	880
S9	890	884	896	898
S10	886	888	886	890

1000-node (density = 0.60%)
Instances on cubic lattice graphs,
modeling Ising spin glasses,
proposed by Burer et al. (2001).

Time w.r.t. CIRCUT

problem	CIRCUT	GPR	GVNSPR	VNSPR
S1	1	5	21	193
S2	1	5	20	180
S3	1	5	20	184
S4	1	6	22	192
S5	1	5	21	181
S6	1	5	19	150
S7	1	5	21	183
S8	1	5	22	190
S9	1	5	19	173
S10	1	5	20	180

2744-node (density = 0.22%)

Instances on cubic lattice graphs, modeling Ising spin glasses, proposed by Burer et al. (2001).

Cut value

problem	CIRCUT	GPR
T1	2410	2378
T2	2416	2382
T3	2408	2390
T4	2414	2382
T5	2406	2374
T6	2412	2390
T7	2410	2384
T8	2418	2378
T9	2388	2362
T10	2420	2390

2744-node (density = 0.22%)

Instances on cubic lattice graphs, modeling Ising spin glasses, proposed by Burer et al. (2001).

Cut value

problem	CIRCUT	GVNSPR
T1	2410	2388
T2	2416	2410
T3	2408	2394
T4	2414	2400
T5	2406	2390
T6	2412	2406
T7	2410	2394
T8	2418	2396
T9	2388	2372
T10	2420	2406

2744-node (density = 0.22%)

Instances on cubic lattice graphs, modeling Ising spin glasses, proposed by Burer et al. (2001).

Cut value

problem	CIRCUT	[redacted]	VNSPR
T1	2410		2416
T2	2416		2416
T3	2408		2406
T4	2414		2418
T5	2406		2416
T6	2412		2420
T7	2410		2404
T8	2418		2418
T9	2388		2384
T10	2420		2422

2744-node (density = 0.22%)
Instances on cubic lattice graphs, modeling Ising spin glasses, proposed by Burer et al. (2001).

Cut value

problem	CIRCUT	GPR	GVNSPR	VNSPR
T1	2410	2378	2388	2416
T2	2416	2382	2410	2416
T3	2408	2390	2394	2406
T4	2414	2382	2400	2418
T5	2406	2374	2390	2416
T6	2412	2390	2406	2420
T7	2410	2384	2394	2404
T8	2418	2378	2396	2418
T9	2388	2362	2372	2384
T10	2420	2390	2406	2422

2744-node (density = 0.22%)

Instances on cubic lattice graphs, modeling Ising spin glasses, proposed by Burer et al. (2001).

Time w.r.t. CIRCUT

problem	CIRCUT	GPR	GVNSPR	VNSPR
T1	1	13	49	493
T2	1	14	54	534
T3	1	13	51	504
T4	1	14	53	558
T5	1	13	49	507
T6	1	15	57	572
T7	1	13	49	493
T8	1	15	56	587
T9	1	15	58	581
T10	1	13	49	502

Concluding remarks

- Randomized heuristics appear to produce better cuts than algorithms based on semidefinite programming (such as DSDP).
- CIRCUT and randomized heuristics are competitive w.r.t. cuts produced, but CIRCUT is much faster.
- CIRCUT may benefit from local search & path-relinking (we are experimenting with a new version of CIRCUT that has a path-relinking phase).
- These slides are available at http://www.research.att.com/~mgcr