GRASP with evolutionary path-relinking for the antibandwidth problem

VIII Metaheuristics International Conference (MIC 2009)
Hamburg, Germany ~ July 13-16, 2009

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

Joint work with A. Duarte, R. Martí, & R. Silva

Summary

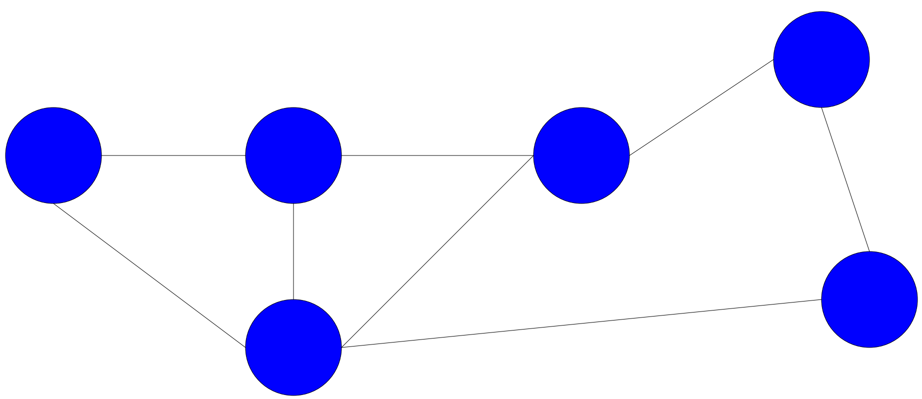
- Antibandwidth
- Integer programming formulation
- GRASP construction
- Local search
- GRASP with evolutionary path-relinking
- Experimental results
- Concluding remarks

Antibandwidth problem

- Given an undirected graph $G = (V,E)$, where
  - V is the set of nodes ($n = |V|$)
  - E is the set of edges ($m = |E|$)
- A labeling f of V is a one-to-one mapping of $\{1,2,\ldots,n\}$ onto V.
  - Each vertex $v \in V$ has a unique label $f(v) \in \{1,2,\ldots,n\}$

Antibandwidth problem

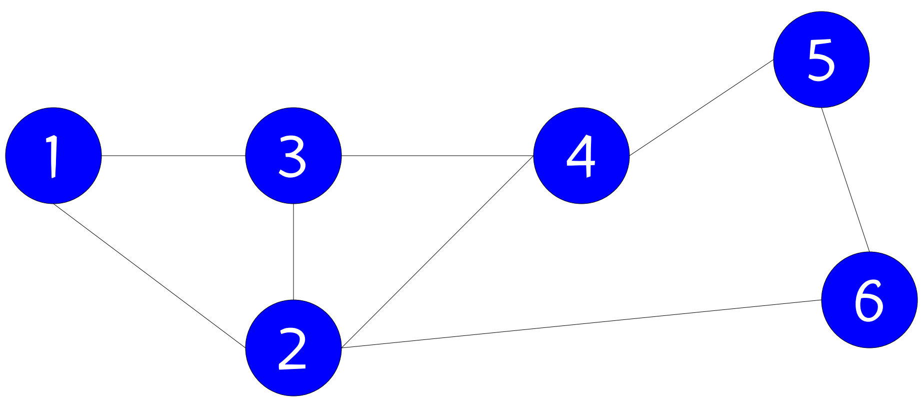
- Given an undirected graph $G = (V,E)$, where
  - V is the set of nodes ($n = |V|$)
  - E is the set of edges ($m = |E|$)
- A labeling f of V is a one-to-one mapping of $\{1,2,\ldots,n\}$ onto V.
  - Each vertex $v \in V$ has a unique label $f(v) \in \{1,2,\ldots,n\}$



undirected graph
$G = (V,E)$

Antibandwidth problem

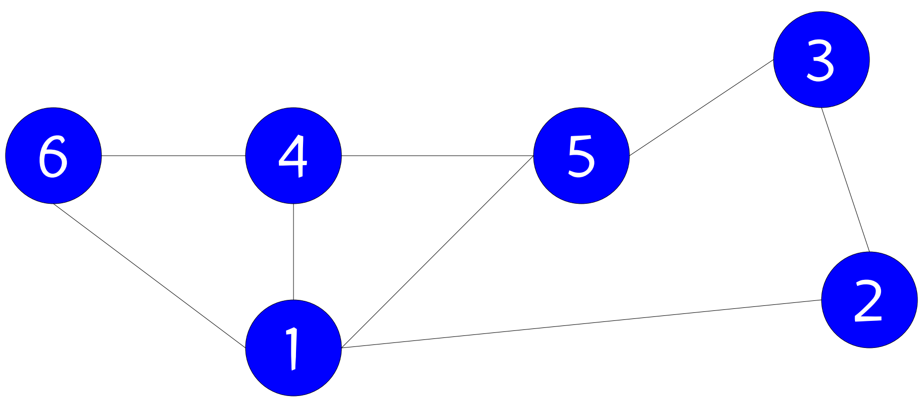
- Given an undirected graph $G = (V,E)$, where
  - V is the set of nodes ($n = |V|$)
  - E is the set of edges ($m = |E|$)
- A labeling f of V is a one-to-one mapping of $\{1,2,\ldots,n\}$ onto V.
  - Each vertex $v \in V$ has a unique label $f(v) \in \{1,2,\ldots,n\}$



undirected graph $G = (V,E)$ with a labeling f

Antibandwidth problem

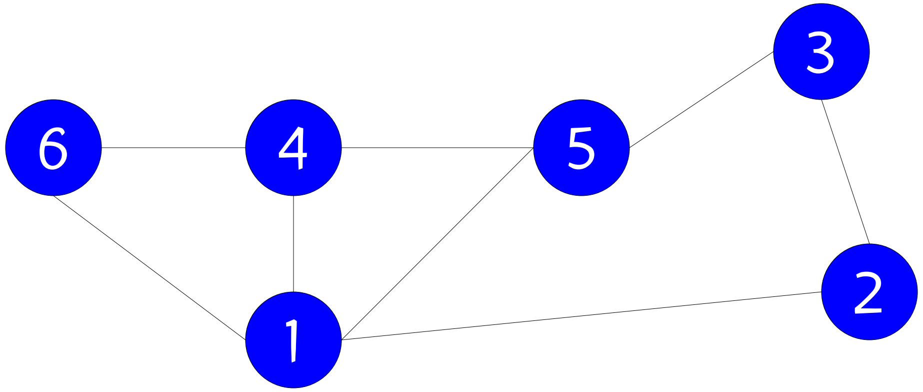
- Given an undirected graph $G = (V,E)$, where
  - V is the set of nodes ($n = |V|$)
  - E is the set of edges ($m = |E|$)
- A labeling f of V is a one-to-one mapping of $\{1,2,\ldots,n\}$ onto V.
  - Each vertex $v \in V$ has a unique label $f(v) \in \{1,2,\ldots,n\}$



undirected graph $G = (V,E)$ with another labeling f

Antibandwidth problem

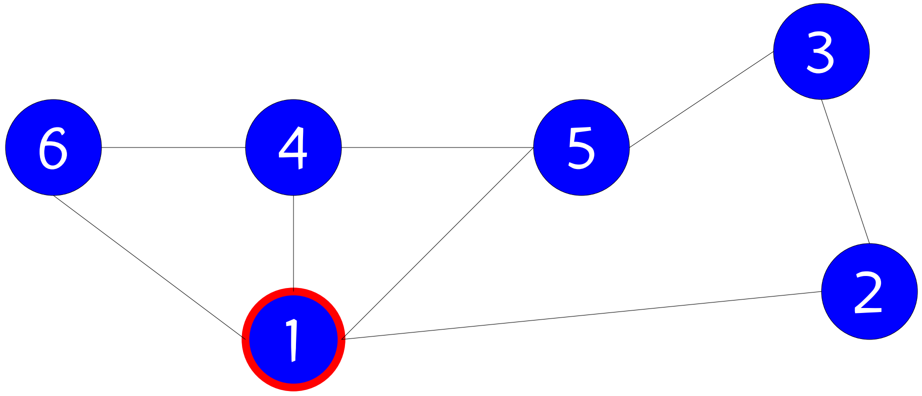
- Given G and f, the antibandwidth $AB_f(v)$ of node v is smallest difference between f(v) and the labels of all of the nodes adjacent to v, i.e.
  - $AB_f(v) = \min \{ \, |f(v) - f(u)| : u \in N(v) \, \}$
  - where N(v) is the set of nodes adjacent to v



undirected graph G = (V,E) with a labeling f

Antibandwidth problem

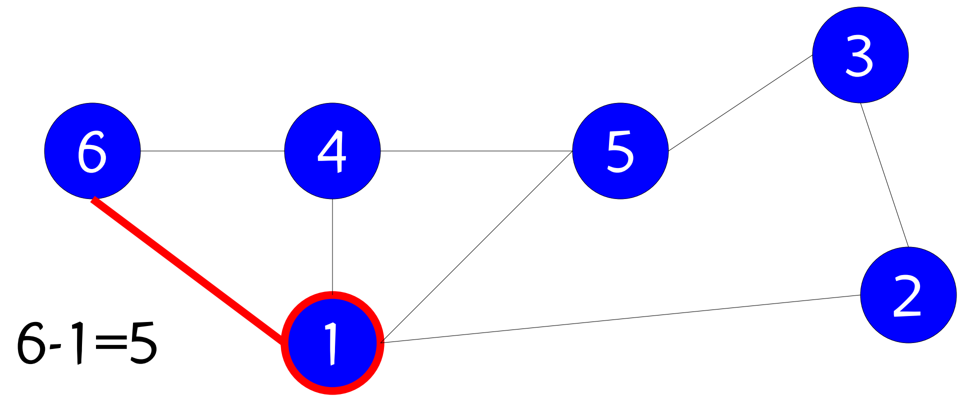
- Given G and f, the antibandwidth $AB_f(v)$ of node v is smallest difference between f(v) and the labels of all of the nodes adjacent to v, i.e.
  - $AB_f(v) = \min \{ \, |f(v) - f(u)| : u \in N(v) \}$
  - where N(v) is the set of nodes adjacent to v



undirected graph G = (V,E) with a labeling f

Antibandwidth problem

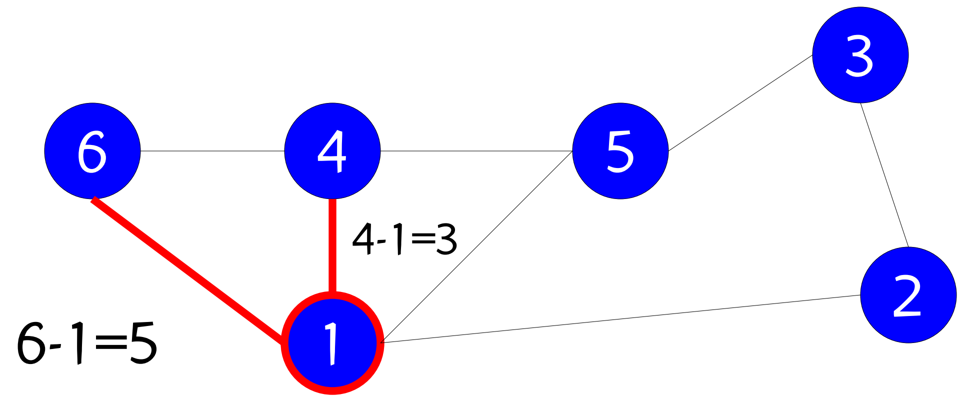
- Given G and f, the antibandwidth $AB_f(v)$ of node v is smallest difference between f(v) and the labels of all of the nodes adjacent to v, i.e.
  - $AB_f(v) = \min \{ \, |f(v) - f(u)| : u \in N(v) \, \}$
  - where N(v) is the set of nodes adjacent to v



undirected graph G = (V,E) with a labeling f

Antibandwidth problem

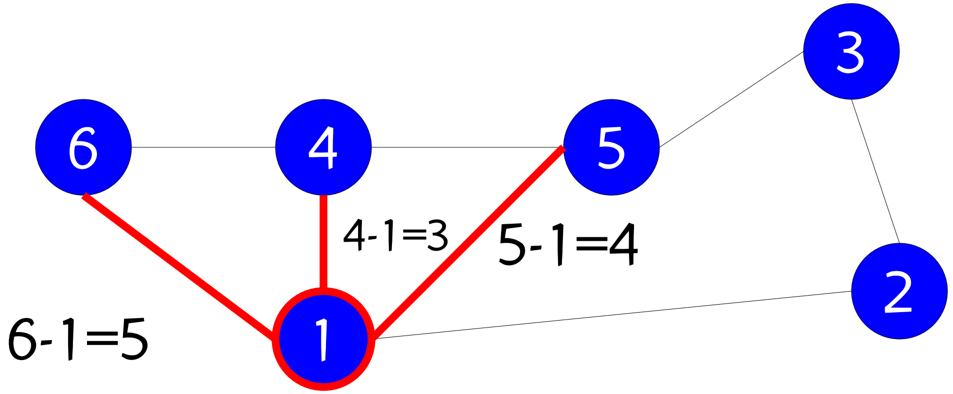
- Given G and f, the antibandwidth $AB_f(v)$ of node v is smallest difference between f(v) and the labels of all of the nodes adjacent to v, i.e.
  - $AB_f(v) = \min \{ \, |f(v) - f(u)| : u \in N(v) \}$
  - where N(v) is the set of nodes adjacent to v



undirected graph G = (V,E) with a labeling f

Antibandwidth problem

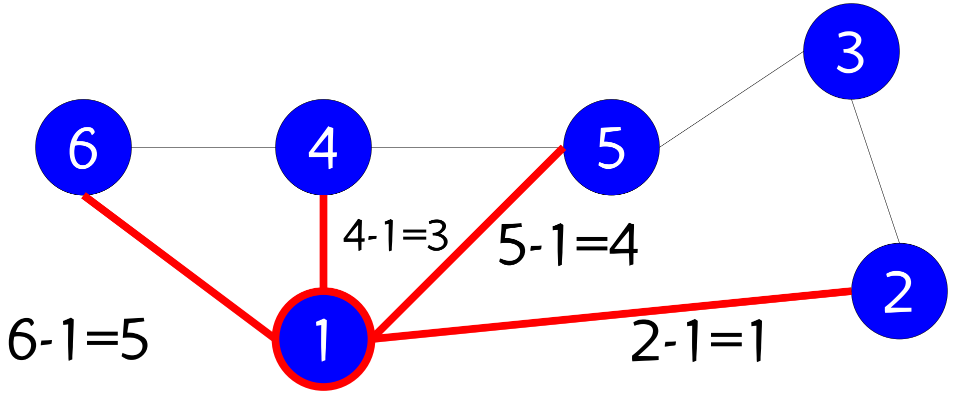
- Given G and f, the antibandwidth $AB_f(v)$ of node v is smallest difference between f(v) and the labels of all of the nodes adjacent to v, i.e.
  - $AB_f(v) = \min \{ \, |f(v) - f(u)| : u \in N(v) \, \}$
  - where N(v) is the set of nodes adjacent to v



undirected graph G = (V,E) with a labeling f

Antibandwidth problem

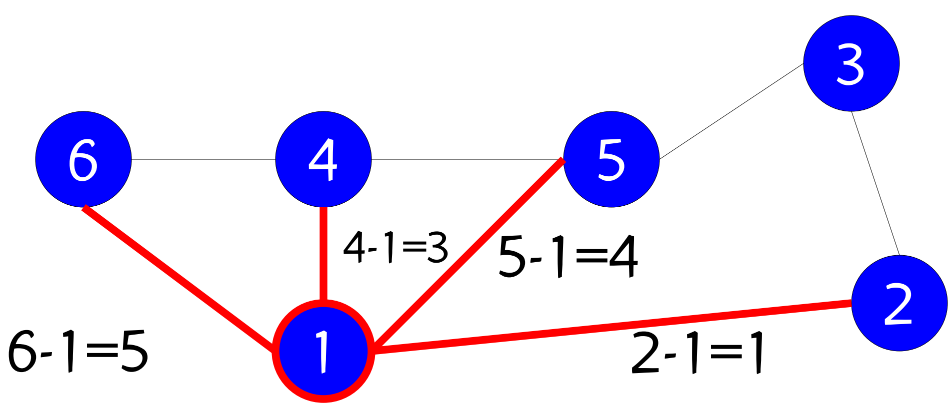
- Given G and f, the antibandwidth $AB_f(v)$ of node v is smallest difference between f(v) and the labels of all of the nodes adjacent to v, i.e.
  - $AB_f(v) = \min \{ \, |f(v) - f(u)| : u \in N(v) \, \}$
  - where N(v) is the set of nodes adjacent to v



undirected graph G = (V,E) with a labeling f

Antibandwidth problem

- Given G and f, the antibandwidth $AB_f(v)$ of node v is smallest difference between f(v) and the labels of all of the nodes adjacent to v, i.e.
  - $AB_f(v) = \min \{ \, |f(v) - f(u)| : u \in N(v) \, \}$
  - where N(v) is the set of nodes adjacent to v

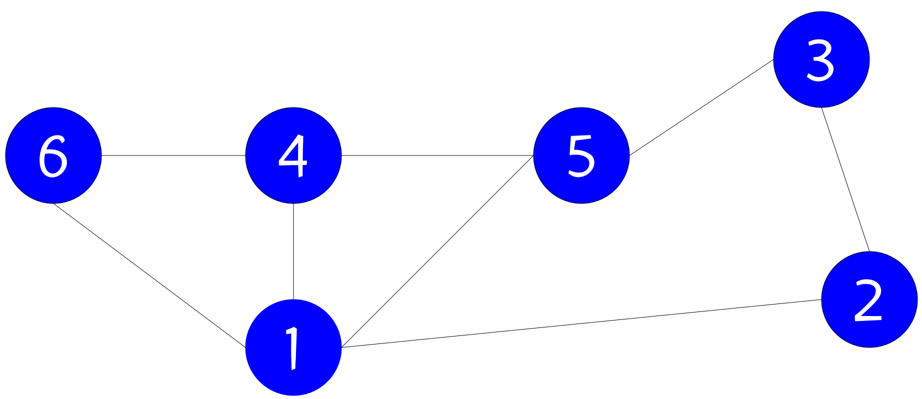


undirected graph G = (V,E) with a labeling f

$AB_f(1) = \min \{5,3,4,1\} = 1$

Antibandwidth problem

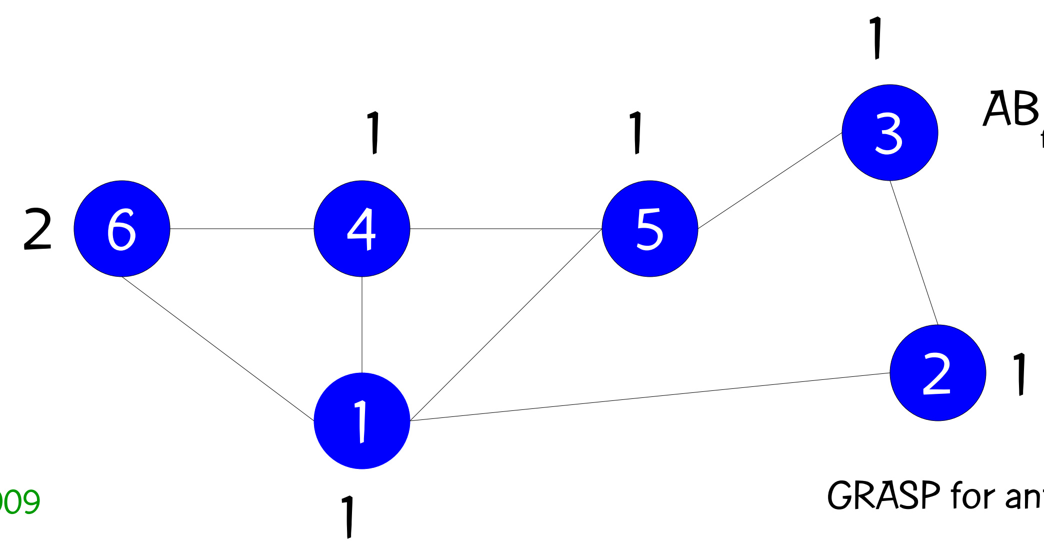
- Given G and f, the antibandwidth $AB_f(G)$ of f is smallest antibandwidth over all nodes in V, i.e.
  - $AB_f(G) = \min \{ AB_f(v) : v \in V \}$
  - where N(v) is the set of nodes adjacent to v



undirected graph G = (V,E) with a labeling f

Antibandwidth problem

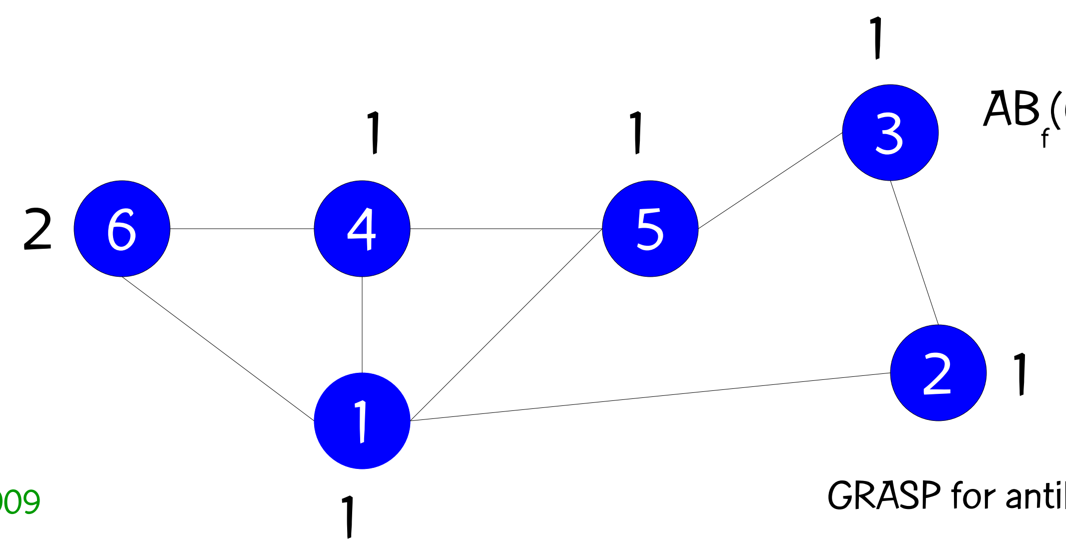
- Given G and f, the antibandwidth $AB_f(G)$ of f is smallest antibandwidth over all nodes in V, i.e.
  - $AB_f(G) = \min \{ AB_f(v) : v \in V \}$
  - where N(v) is the set of nodes adjacent to v



$AB_f(G) = \min \{ 1, 1, 1, 1, 2 \} = 1$

Antibandwidth problem

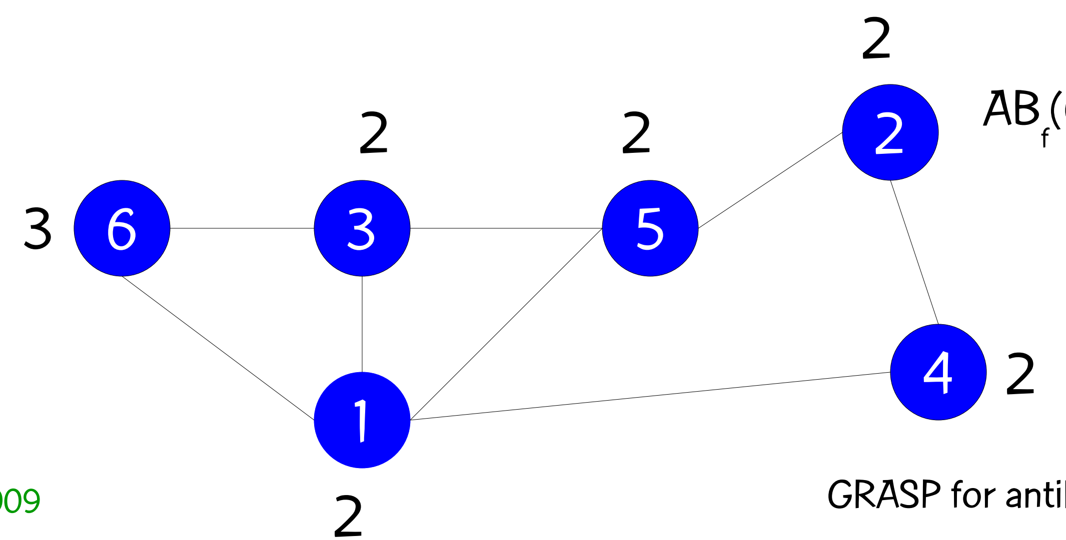
- Given G, the antibandwidth AB(G) of G is largest antibandwidth over all possible labelings, i.e.
  - $AB(G) = \max \{ AB_f(G) : f \in \Pi_n \}$
  - where Π_n is the set of all permutations of $\{1, 2, \ldots, n\}$



$AB_f(G) = \min \{ 1, 1, 1, 1, 2 \} = 1$

Antibandwidth problem

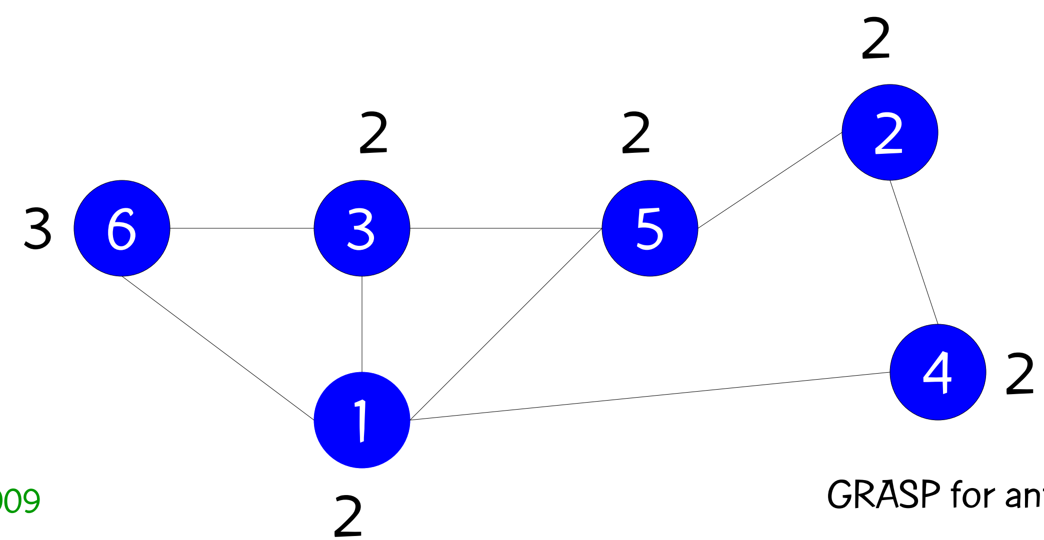
- Given G, the antibandwidth AB(G) of G is largest antibandwidth over all possible labelings, i.e.
  - $AB(G) = \max \{ AB_f(G) : f \in \Pi_n \}$
  - where Π_n is the set of all permutations of $\{1, 2, \ldots, n\}$



$AB_f(G) = \min \{ 2, 2, 2, 2, 3 \} = 2$

Antibandwidth problem

- Given G, the antibandwidth AB(G) of G is largest antibandwidth over all possible labelings, i.e.
  - $AB(G) = \max \{ AB_f(G) : f \in \Pi_n \}$
  - where Π_n is the set of all permutations of {1, 2, ..., n}



AB(G) = 2

Antibandwidth problem

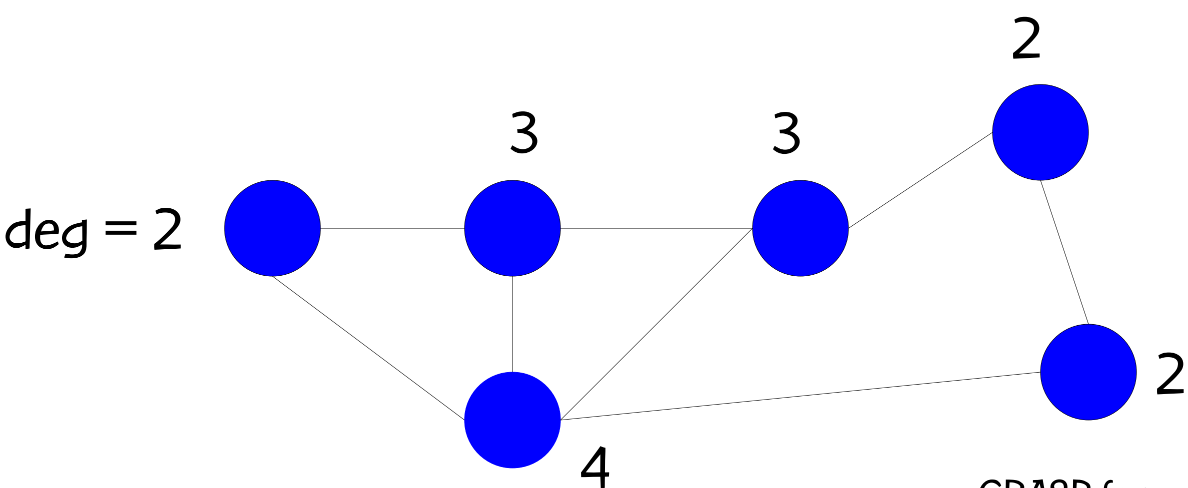
- NP-hard (Leung et al., 1984)
- Special cases can be solved in polynomial time, e.g. complements of intervals, arborescent comparability, and on threshold graphs (Raspaud et al., 2008)

Antibandwidth problem

- Yixum and Jinjiang (2003) proposed the upper bound: min { floor($(n - \delta + 1)/2$), $n - \Delta$ }, where
  - δ is the smallest degree over all $v \in V$
  - Δ is the largest degree over all $v \in V$

Antibandwidth problem

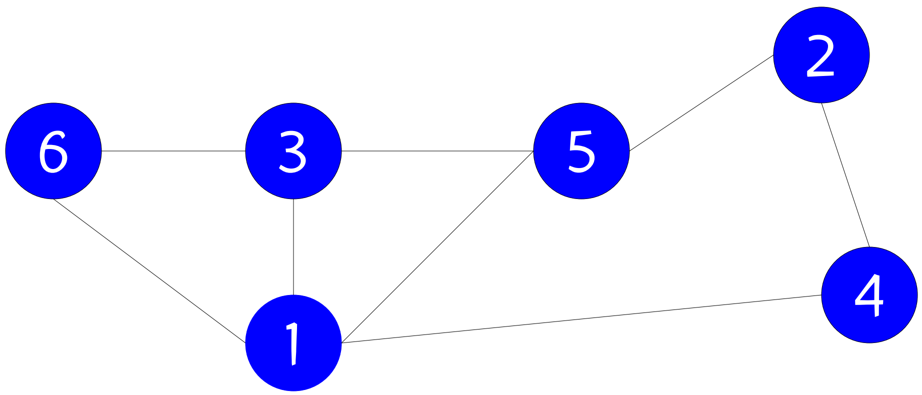
- Yixum and Jinjiang (2003) proposed the upper bound: min { floor((n – δ + 1)/2), n – Δ }, where
  - δ is the smallest degree over all v $\in$ V
  - Δ is the largest degree over all v $\in$ V



$\delta = 2$
$\Delta = 4$
UB = min{ 2, 2 } = 2

Antibandwidth problem

- Yixum and Jinjiang (2003) proposed the upper bound: min { floor($(n - \delta + 1)/2$), $n - \Delta$ }, where
  - δ is the smallest degree over all $v \in V$
  - Δ is the largest degree over all $v \in V$



$\delta = 2$

$\Delta = 4$

UB = min{ 2, 2 } = 2

$AB_f(G) = 2$ is opt

$AB(G) = 2$

Antibandwidth problem: IP formulation

- Let x_{ik} be a binary variable that takes on the value 1 if and only if $f(i) = k$, i.e. node i takes label k.
- Define $l_i = f(i) \in \{1, 2, \ldots, n\}$ to be the label of node i.
- Finally, let $b = AB_f(G) = \min\{|f(u) - f(v)| : (u,v) \in E\}$ be the antibandwidth of labeling f.
- In the antibandwidth problem we want to determine the labeling f^* that maximizes b.

Antibandwidth problem: IP formulation

- Objective: maximize *b*
- Constraints:
  - One label is assigned to each node:

$$\sum_{i=1}^{n} x_{ik} = 1, \quad \forall\, k = 1,\ldots,n$$

Antibandwidth problem: IP formulation

- Objective: maximize *b*
- Constraints:
  - One node is assigned to each label:

$$\sum_{k=1}^{n} x_{ik} = 1, \quad \forall\, i = 1,\ldots,n$$

Antibandwidth problem: IP formulation

- Objective: maximize *b*
- Constraints:
  - Each label l_i is a function of the binary variables x_{ik}:

$$\sum_{k=1}^{n} k \cdot x_{ik} = l_i, \quad \forall\, i = 1, \ldots, n$$

Antibandwidth problem: IP formulation

- Objective: maximize *b*
- Constraints:
  - Require that $b = \min\{|l_i - l_j| : (i,j) \in E\}$:

$$b \le |l_i - l_j|, (i,j) \in E$$

Antibandwidth problem: IP formulation

- Objective: maximize *b*
- Constraints:
  - Binary variables x_{ik} can only take values 0 or 1 :

$$x_{ik} \in \{0,1\}, \forall\, i,k = 1,\ldots,n$$

Antibandwidth problem: IP formulation

- Objective: maximize *b*
- Constraints:
  - Labels l_i can only take on values {1, ..., n} :

$$l_i \in \{1,2,\ldots,n\}, \forall\, i = 1,\ldots,n$$

Antibandwidth problem: IP formulation

- Objective: maximize b
- Constraints $b \le |l_i - l_j|, (i,j) \in E$ are nonlinear:
  - If $l_i \ge l_j$ then $b \le l_i - l_j$
  - Otherwise, $b \le -(l_i - l_j)$
  - Introduce two binary variables to indicate case:
    - If $l_i \ge l_j$ then $y_{ij} = 0$ and $z_{ij} = 1$
    - Otherwise, $y_{ij} = 1$ and $z_{ij} = 0$

Antibandwidth problem: IP formulation

- Objective: maximize b
- Constraints $b \le |l_i - l_j|, (i, j) \in E$ become:

$$b - (l_i - l_j) \le 2y_{ij}(n - 1), \forall (i, j) \in E$$

$l_i \ge l_j$	then	$y_{ij} = 0$	$z_{ij} = 1$
$l_i < l_j$	then	$y_{ij} = 1$	$z_{ij} = 0$

$$b + (l_i - l_j) \le 2z_{ij}(n - 1), \forall (i, j) \in E$$

$$y_{ij} + z_{ij} = 1, \forall (i, j) \in E$$

$$b \ge 1$$

Antibandwidth problem: IP formulation

$$\max \quad b$$

$$\sum_{i=1}^{n} x_{ik} = 1, \quad \forall\, k = 1,\ldots,n$$

$$\sum_{k=1}^{n} x_{ik} = 1, \quad \forall\, i = 1,\ldots,n$$

$$\sum_{k=1}^{n} k \cdot x_{ik} = l_i, \quad \forall\, i = 1,\ldots,n$$

$$b - (l_i - l_j) \le 2y_{ij}(n-1), \forall\, (i,j) \in E$$

$$b + (l_i - l_j) \le 2z_{ij}(n-1), \forall\, (i,j) \in E$$

$$y_{ij} + z_{ij} = 1, \forall\, (i,j) \in E$$

$$b \ge 1$$

$$x_{ik} \in \{0,1\}, \forall\, (i,k) \in E$$

$$l_i \in \{1,2,\ldots n\}, \forall\, i = 1,2,\ldots n$$

$$y_{ik} \in \{0,1\}, \forall\, (i,k) \in E$$

$$z_{ik} \in \{0,1\}, \forall\, (i,k) \in E$$

Antibandwidth problem: IP formulation

$$\max \quad b$$

$$\sum_{i=1}^{n} x_{ik} = 1, \quad \forall\, k = 1,\ldots,n$$

$$\sum_{k=1}^{n} x_{ik} = 1, \quad \forall\, i = 1,\ldots,n$$

$$\sum_{k=1}^{n} k \cdot x_{ik} = l_i, \quad \forall\, i = 1,\ldots,n$$

$$b - (l_i - l_j) \le 2y_{ij}(n-1), \forall\, (i,j) \in E$$

$$b + (l_i - l_j) \le 2z_{ij}(n-1), \forall\, (i,j) \in E$$

$$y_{ij} + z_{ij} = 1, \forall\, (i,j) \in E$$

$$b \ge 1$$

$$x_{ik} \in \{0,1\}, \forall\, (i,k) \in E$$

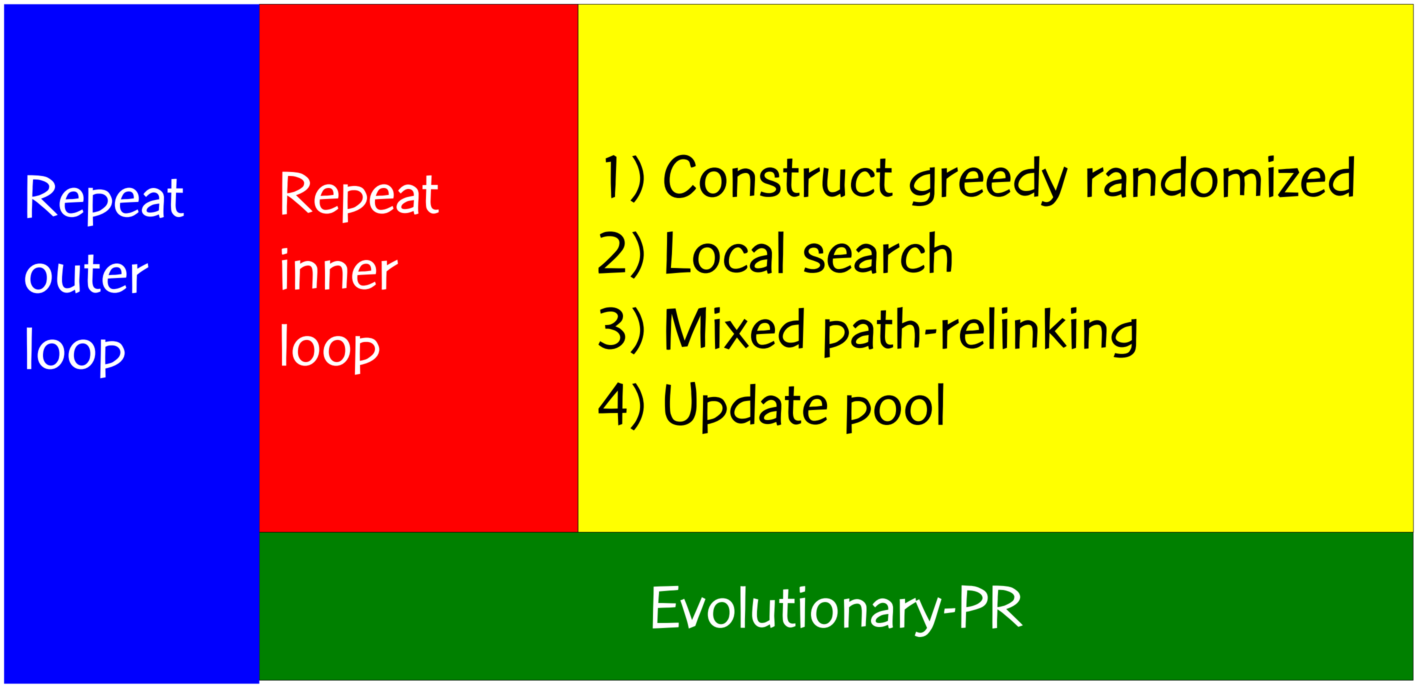
$$l_i \in \{1,2,\ldots n\}, \forall\, i = 1,2,\ldots n$$

$$y_{ik} \in \{0,1\}, \forall\, (i,k) \in E$$

$$z_{ik} \in \{0,1\}, \forall\, (i,k) \in E$$

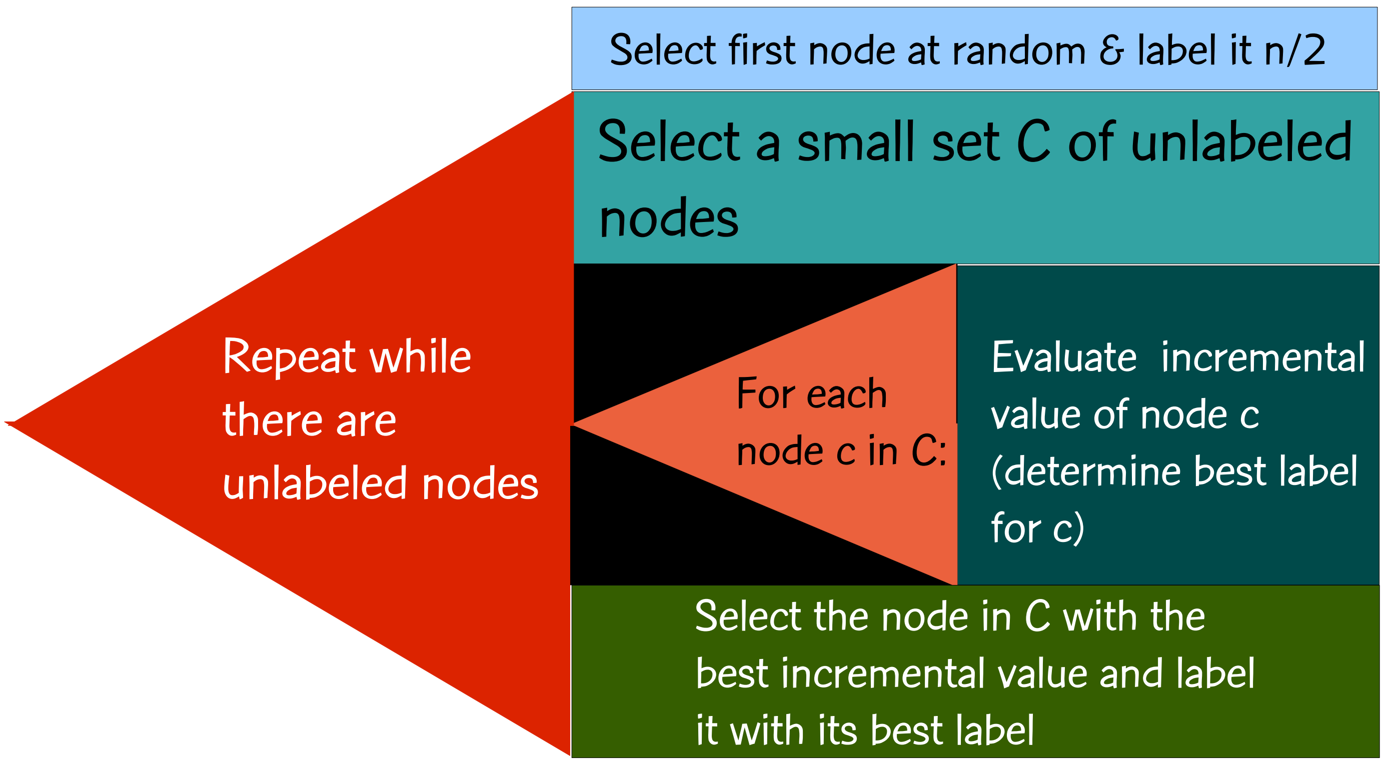
IP has $O(n^2)$ variables and $O(n^2)$ constraints

GRASP with evolutionary path-relinking



GRASP construction procedure

- We use the sampled greedy construction scheme of R. & Werneck (2004)



GRASP construction procedure:

Selecting a small set C of unlabeled nodes

- The set CL of candidate nodes is made up of nodes adjacent to labeled nodes
- The small set C of candidate nodes is a set of $\alpha \times |CL|$ randomly sampled nodes from CL, where α is a random real number $\in [0,1]$
- The value of α does not change during construction
- Values of $\alpha \approx 1$ makes sampled greedy resemble a greedy construction, while values of $\alpha \approx 0$ makes it behave like a random construction

GRASP construction procedure:

Determine the best label for a candidate node c

- Let $\breve{l}_c$ and $\widehat{l}_c$ be, respectively, the smallest and largest assigned labels to the the nodes adjacent to c
- The "best" label for c is

$$l^* = \operatorname{argmax}\{\min(|l - \widehat{l}_c|, |l - \breve{l}_c|) : l = 1, \ldots, n\}$$

- The closest available label to l^* is assigned to c

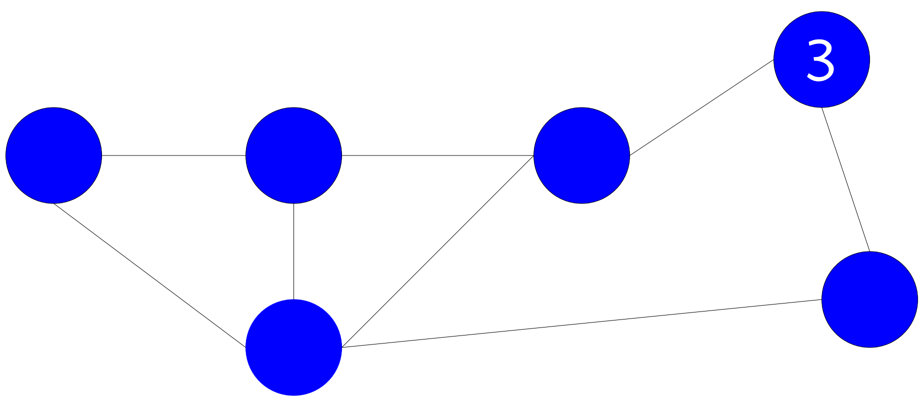
GRASP construction procedure:

Determine the best label for a candidate node

- Let $\breve{l}_c$ and $\overset{\frown}{l}_c$ be, respectively, the smallest and largest assigned labels to the the nodes adjacent to c
- The "best" label for c is

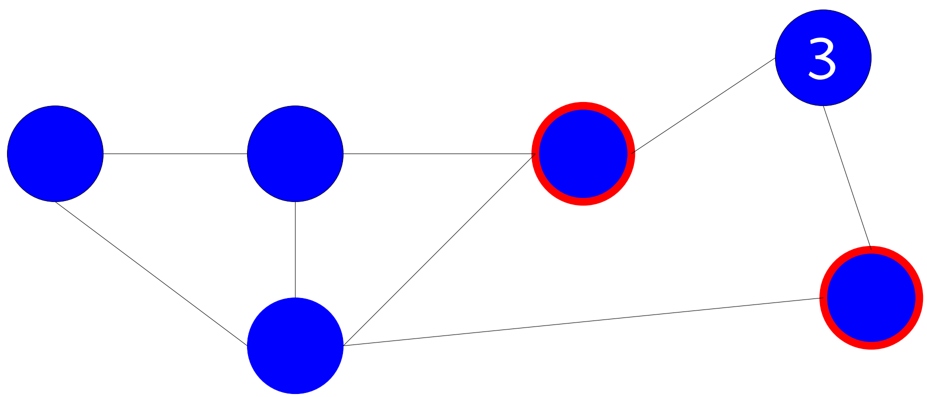
$$l^* = \operatorname{argmax}\{\min(|l - \overset{\frown}{l}_u|, |l - \breve{l}_u|) : l = 1, \ldots, n\}$$

- The closest available label to l^* is assigned to c



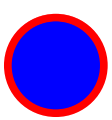
Choose first node at random and label it n/2 = 6/2 = 3

GRASP construction procedure

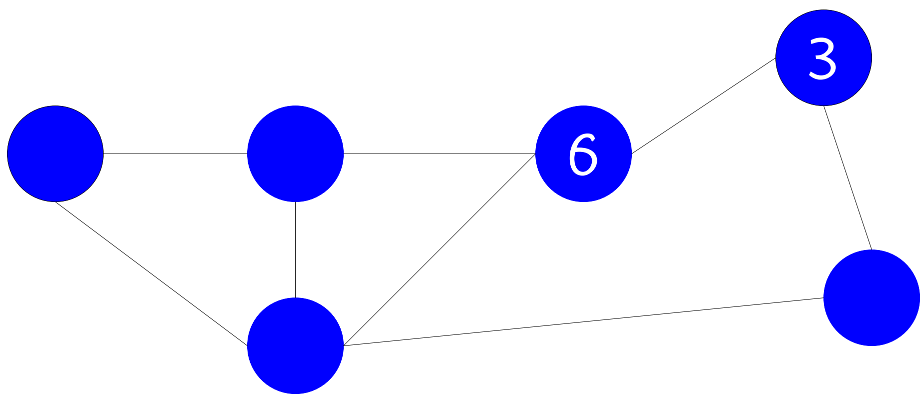


Best label for both candidates is 6.

Label one of the nodes with a 6

 Candidate node

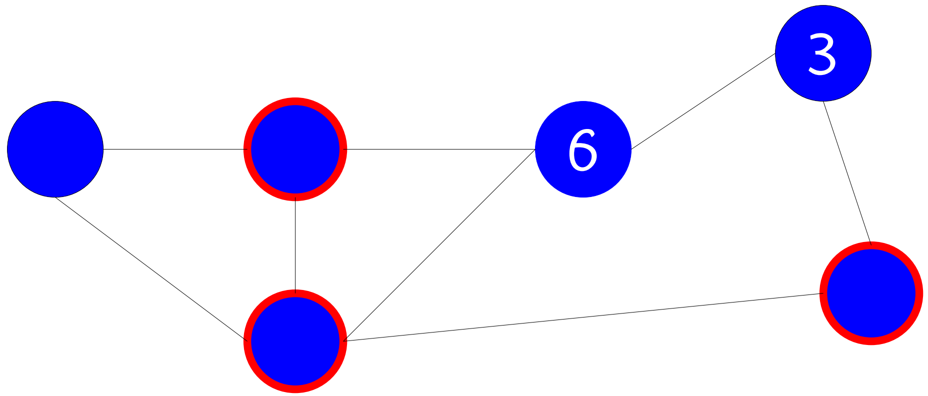
GRASP construction procedure



Best label for both candidates on left is 1 and on right is 6.

Label node on right with a 5 (closest available label to 6)

GRASP construction procedure

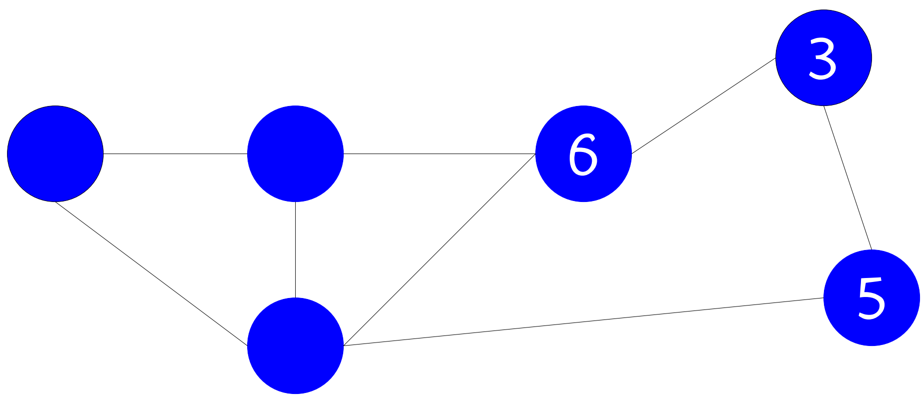


Best label for both candidates on left is 1 and on right is 6.

Label node on right with a 5 (closest available label to 6)

Candidate node

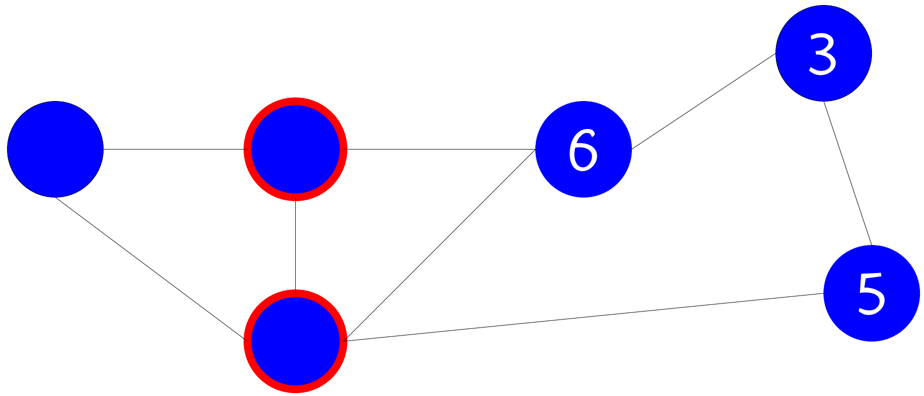
GRASP construction procedure



Best label for both candidates on left is 1 and on right is 6.

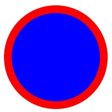
Label node on right with a 5 (closest available label to 6)

GRASP construction procedure



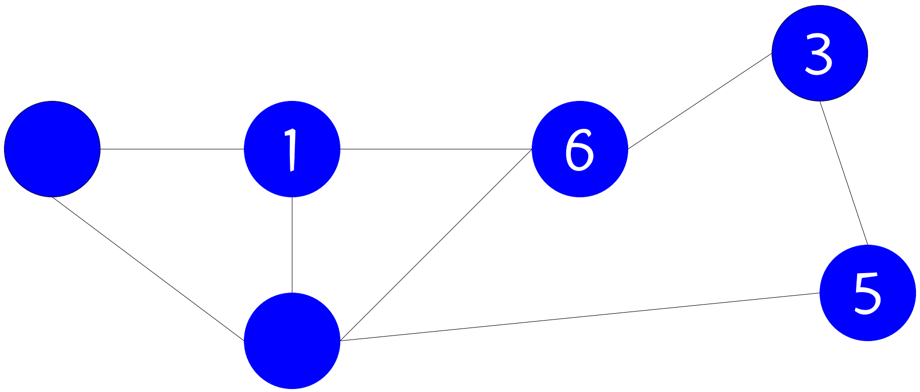
Best label for both candidates is 1.

Label node on top with a 1.



Candidate node

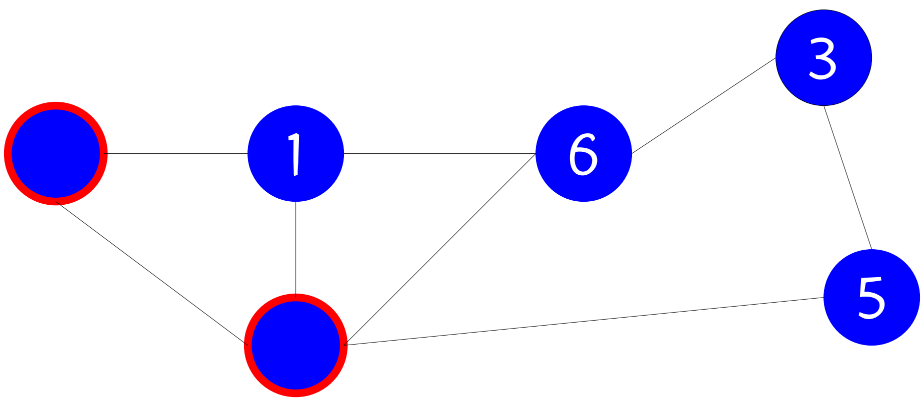
GRASP construction procedure



Best label for both candidates is 1.

Label node on top with a 1.

GRASP construction procedure

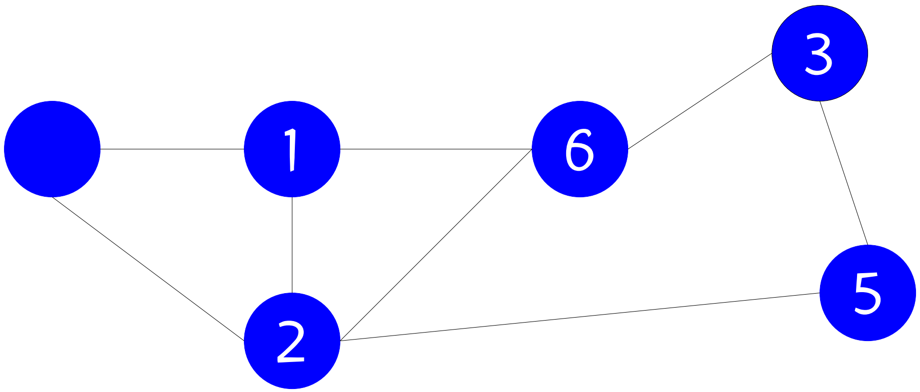


Best label for node on left is 6 and for node on bottom is 3.

Label node on bottom with a 2.

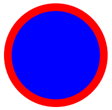
Candidate node

GRASP construction procedure

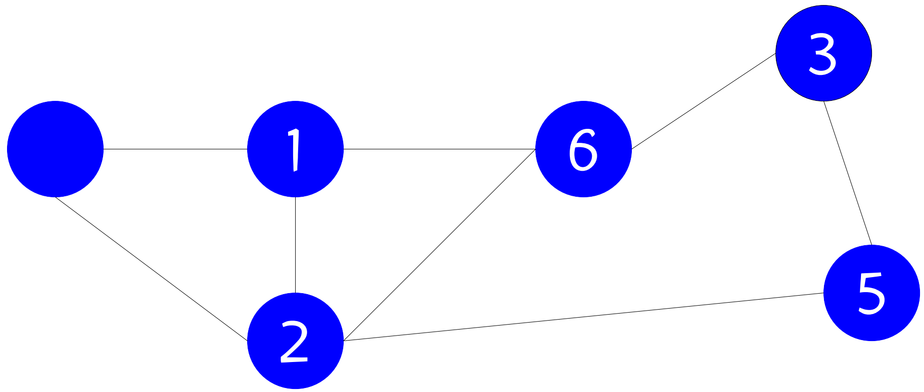


Best label for node on left is 6 and for node on bottom is 2 or 3.

Label node on bottom with a 2.

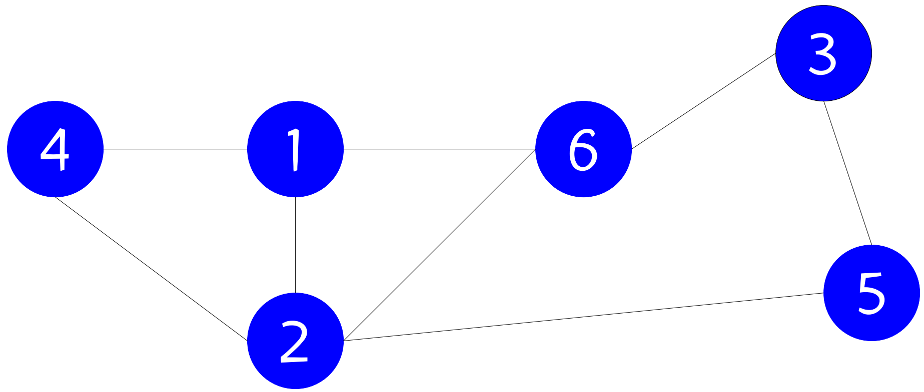
 Candidate node

GRASP construction procedure



Remaining node must be labeled with a 4.

GRASP construction procedure



Remaining node must be labeled with a 4.

$AB_f(G) = 1$

Local Search

GRASP local search procedure

- Antibandwidth problem has a flat landscape: many solutions have same cost
- For a given labeling f, there may be multiple nodes u such that $AB_f(u) = AB_f(G)$
- Therefore, in local search, a move (swap of labels of a pair of nodes) that improves $AB_f(u)$ does not necessarily change the value of the solution $AB_f(G)$

GRASP local search procedure

- Nodes u with unequal $AB_f(u)$ values but that are close to $AB_f(G)$ can be crucial in future iterations (swaps) of the local search, even though they cannot affect the value of the current labeling
- Define the set of crucial vertices of a labeling f to be

$$C(f) = \{u \in V : AB_f(u) \leq \beta \cdot AB_f(G)\}$$

$$(1 \leq \beta \leq 2)$$

at&t Your world. Delivered.

GRASP local search procedure

- Given a labeling f, operator move(u,v) assigns the label $f(u)$ to node v and the label $f(v)$ to node u, resulting in a new labeling f'
- Local search scans nodes u in $C(f)$, changing their labels to increase their antibandwidths
- Let $\breve{l}_u$ and $\hat{l}_u$ be, respectively, the smallest and largest assigned labels to the the nodes adjacent to u
- The best label for u is

$$l_u^* = \operatorname{argmax}\{\min(|l - \hat{l}_u|, |l - \breve{l}_u|) : l = 1, \ldots, n\}$$

GRASP local search procedure

- Once we determine the best label l^* for u, we determine the node v with this label to evaluate move(u,v)
- We know that label l^* is good for u, but we need to determine whether label $f(u)$ is good for node v
- We extend the search for a good label for u not only to node v with label l^*, but also to nodes with labels close to l^*
- The set $N'(u)$ of suitable swapping nodes for u depends on the relationship between l^*, $\breve{l}_u$, and $\widehat{l}_u$

at&t
Your world. Delivered.

GRASP local search procedure

- If $l_u^* < \breve{l}_u$ then $N'(u) = \{v \in V : l_u^* \leq f(v) \leq \breve{l}_u - AB_f(G)\}$
- If $l_u^* > \frown{l}_u$ then $N'(u) = \{v \in V : \frown{l}_u + AB_f(G) \leq f(v) \leq l_u^*\}$
- If $\breve{l}_u \leq l_u^* \leq \frown{l}_u$ then

$$N'(u) = \{v \in V : \frown{l}_u + AB_f(G) \leq f(v) \leq \frown{l}_u - AB_f(G)\}$$

GRASP local search procedure

- If $l_u^* < \breve{l}_u$ then $N'(u) = \{v \in V : l_u^* \leq f(v) \leq \breve{l}_u - AB_f(G)\}$

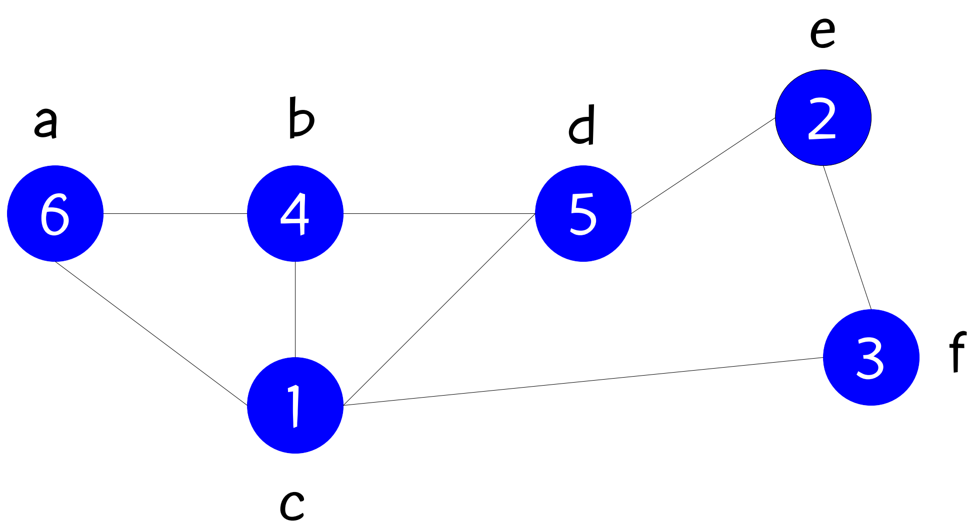
- If $l_u^* > \frown{l}_u$ then $N'(u) = \{v \in V : \frown{l}_u + AB_f(G) \leq f(v) \leq l_u^*\}$

- If $\breve{l}_u \leq l_u^* \leq \frown{l}_u$ then

$$N'(u) = \{v \in V : \breve{l}_u + AB_f(G) \leq f(v) \leq \frown{l}_u - AB_f(G)\}$$

If $N'(u) = \emptyset$, then $AB_f(u)$ cannot be increased in a single step by changing the current label of u.

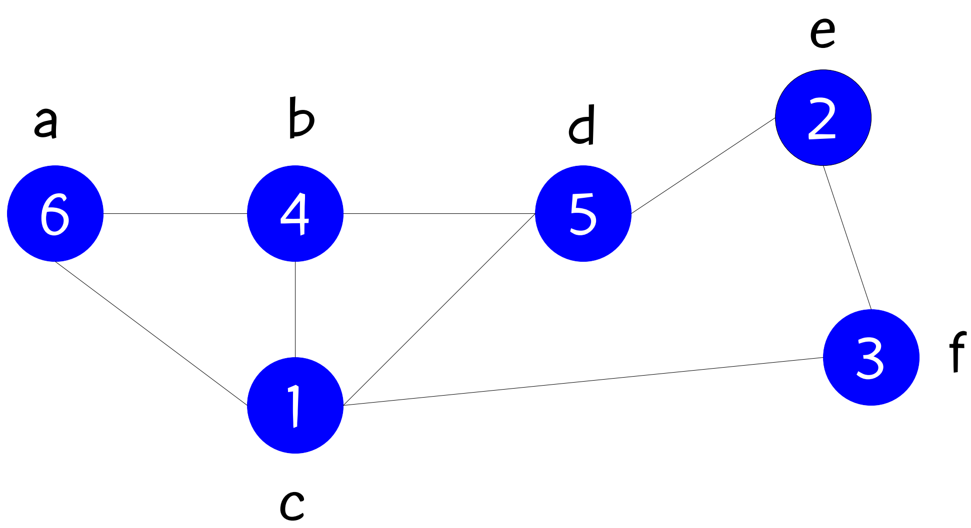
GRASP local search procedure



$AB_f(G) = 1$

v	$AB_f(v)$
a	2
b	1
c	2
d	1
e	1
f	1

GRASP local search procedure

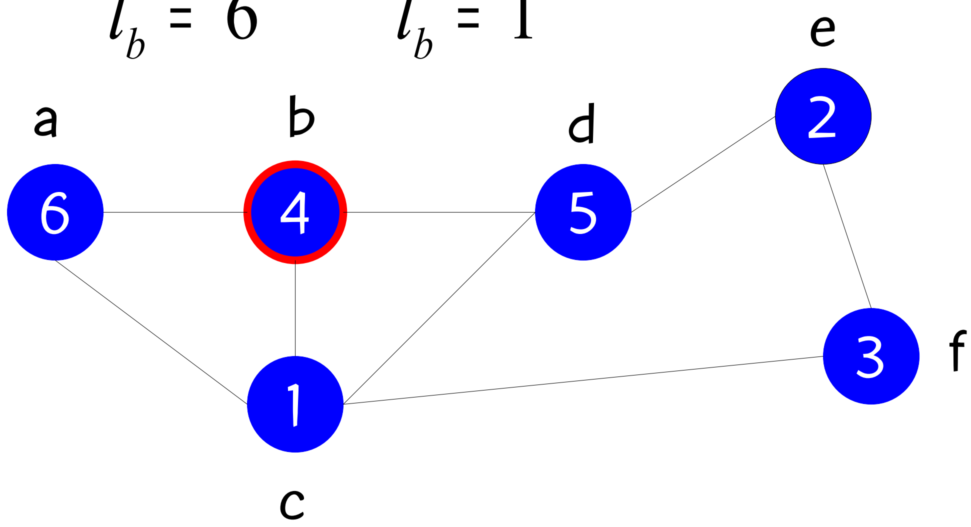


$AB_f(G) = 1$

v	$AB_f(v)$
a	2
b	1 crucial
c	2
d	1 crucial
e	1 crucial
f	1 crucial

GRASP local search procedure

$\hat{l}_b = 6$ $\check{l}_b = 1$

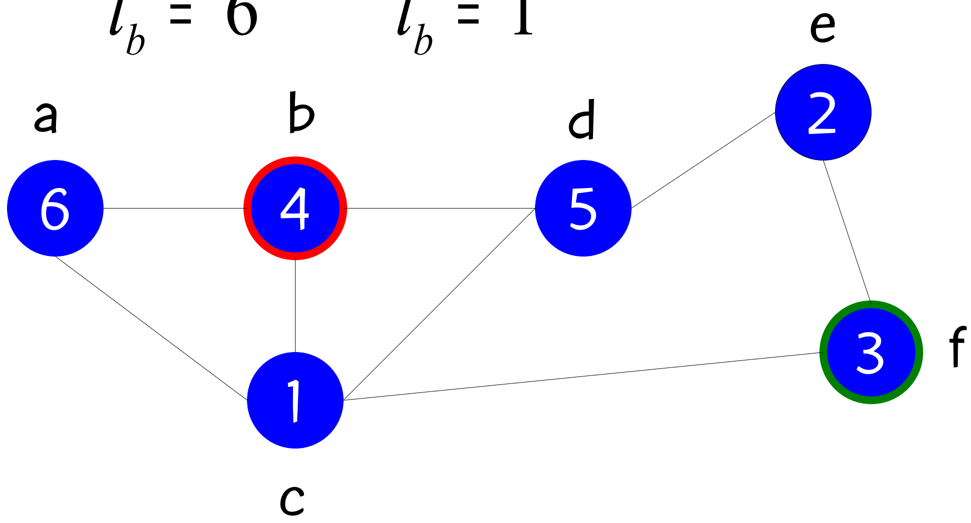


$AB_f(G) = 1$

l	$\lvert l - \hat{l}_u \rvert$	$\lvert l - \check{l}_u \rvert$
1	5	0
2	4	1
3	3	2
4	2	3
5	1	4
6	0	5

GRASP local search procedure

$\widehat{l}_b = 6 \qquad \breve{l}_b = 1$



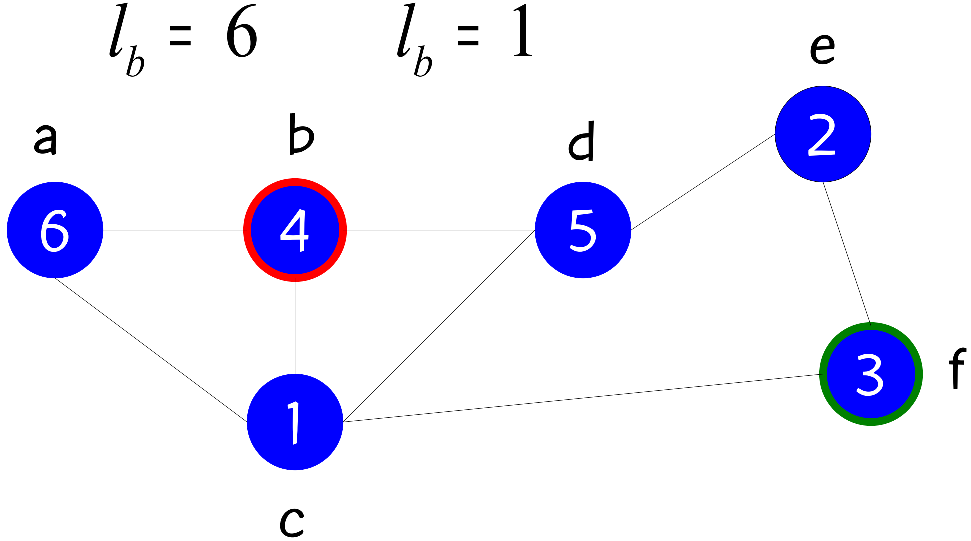
$$l_u^* = \operatorname{argmax}\{\min(|l - \widehat{l}_u|, |l - \breve{l}_u|) : l = 1, \ldots, n\} = 3$$

$AB_f(G) = 1$

l	$\lvert l - \widehat{l}_u \rvert$	$\lvert l - \breve{l}_u \rvert$
1	5	0
2	4	1
3	3	2
4	2	3
5	1	4
6	0	5

GRASP local search procedure

$\hat{l}_b = 6 \qquad \breve{l}_b = 1$



$AB_f(G) = 1$

l	$\lvert l - \hat{l}_u \rvert$	$\lvert l - \breve{l}_u \rvert$
1	5	0
2	4	1
3	3	2
4	2	3
5	1	4
6	0	5

$$l_u^* = \operatorname{argmax}\{\min(|l - \hat{l}_u|, |l - \breve{l}_u|) : l = 1, \ldots, n\} = 3$$

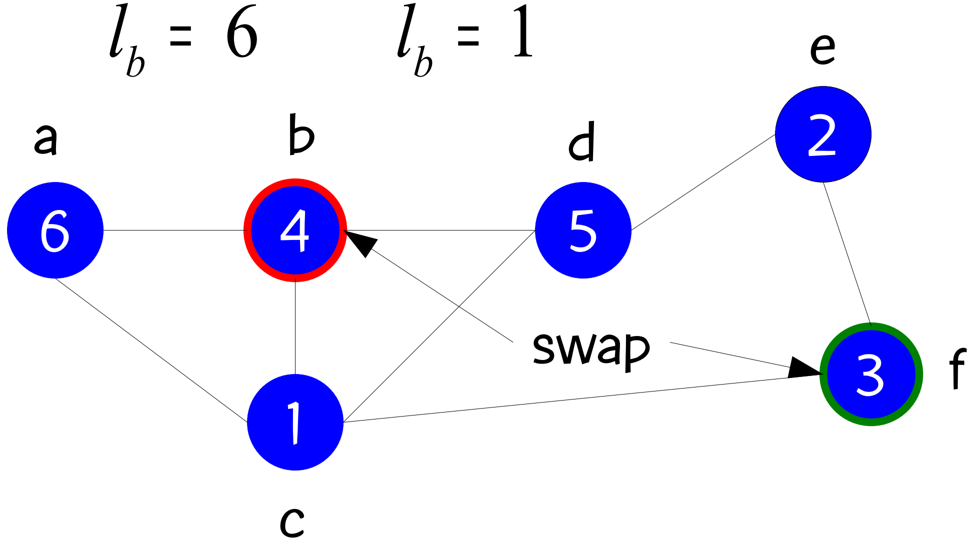
Since $1 = \breve{l}_u \le l_u^* \le \hat{l}_u = 6$ then

$$N'(u) = \{v \in V : \breve{l}_u + AB_f(G) \le f(v) \le \hat{l}_u - AB_f(G)\} = \{d, e, f\}$$

at&t Your world. Delivered.

GRASP local search procedure

$\hat{l}_b = 6 \qquad \breve{l}_b = 1$



$AB_f(G) = 1$

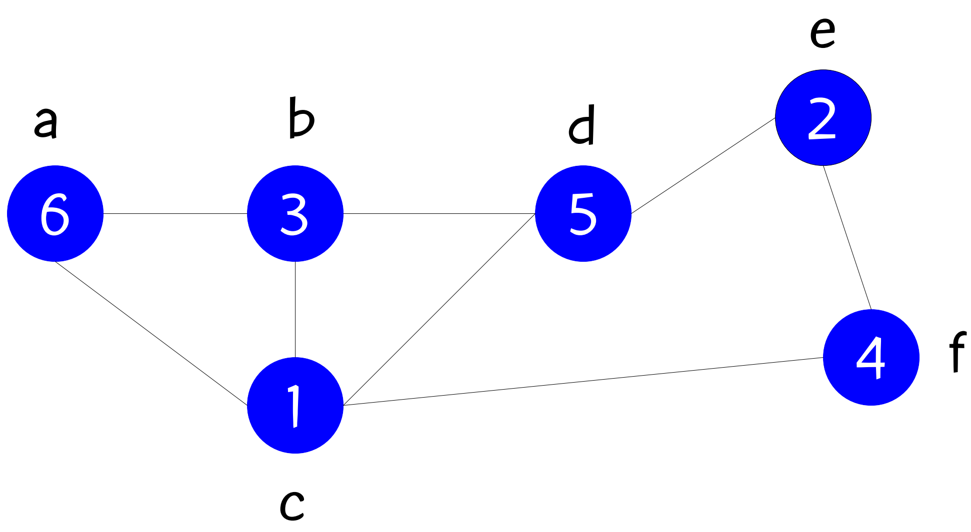
l	$\lvert l - \hat{l}_u \rvert$	$\lvert l - \breve{l}_u \rvert$
1	5	0
2	4	1
3	3	2
4	2	3
5	1	4
6	0	5

$$l_u^* = \operatorname{argmax}\{\min(\lvert l - \hat{l}_u \rvert, \lvert l - \breve{l}_u \rvert) : l = 1,\ldots,n\} = 3$$

Since $1 = \breve{l}_u \le l_u^* \le \hat{l}_u = 6$ then

$$N'(u) = \{v \in V : \breve{l}_u + AB_f(G) \le f(v) \le \hat{l}_u - AB_f(G)\} = \{d, e, f\}$$

GRASP local search procedure



$AB_f(G) = 2$

optimal!

GRASP local search procedure

- Value of a move:
  - Common practice is to define it as change in objective function value
  - In antibandwidth, change in objective function provides little information
- Given node u and node $v \in C(u)$, we define value of move(u,v) to be the difference in the antibandwidth of u.

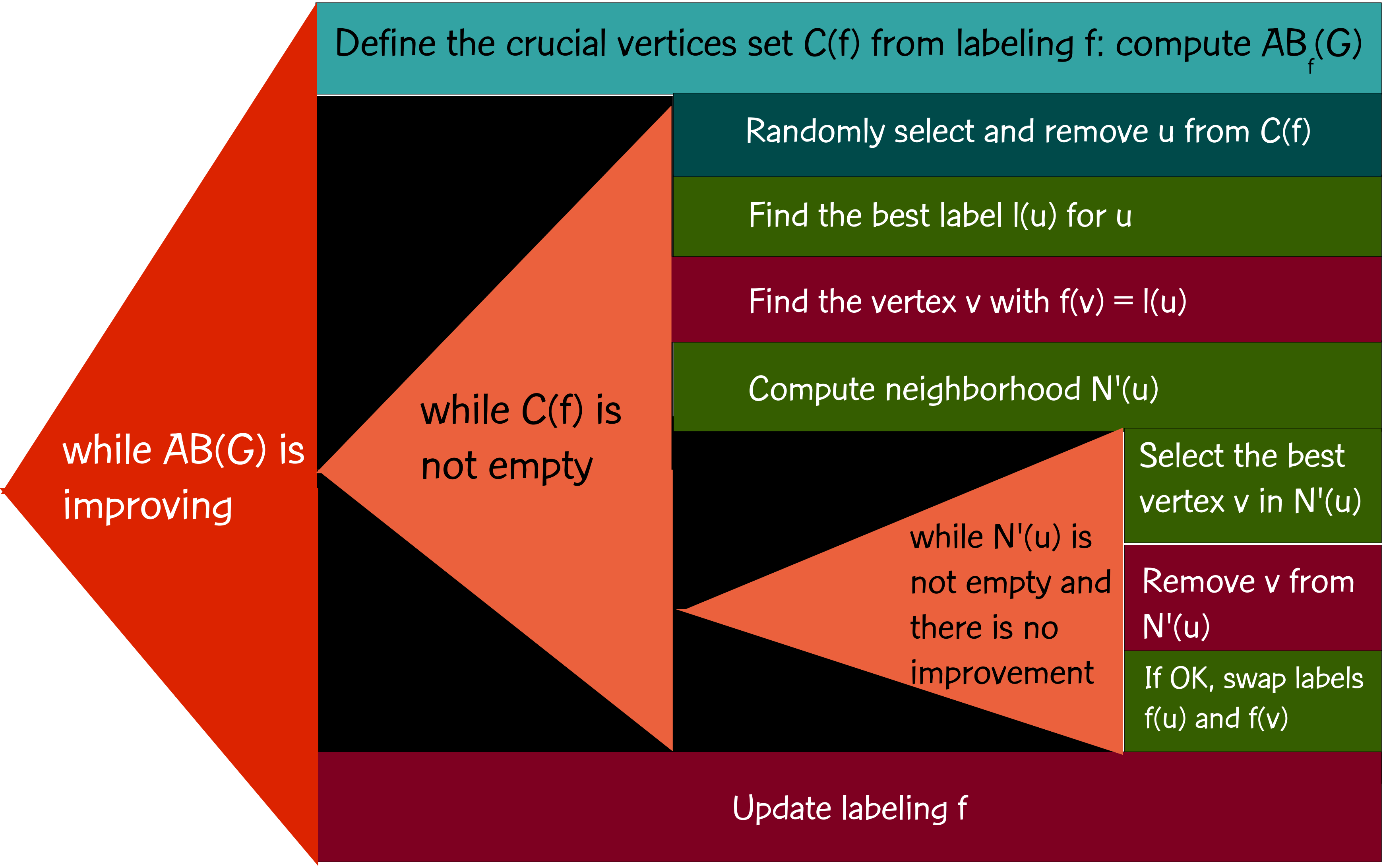
GRASP local search procedure

- If f is the original labeling and f' is the resulting labeling after move(u,v), then

$$\text{moveValue}(u,v) = AB_{f'}(u) - AB_f(u)$$

- Perform move(u,v) only if moveValue(u,v) > 0 and $AB_{f'}(v) \geq AB_f(G)$
- Computation of $AB_f(G)$ is expensive: requires examination of all vertices in graph
  - $AB_f(G)$ is not updated after each move, only when C(f) is computed (a la Glover & Laguna (1997))

GRASP local search procedure



GRASP with evolutionary path-relinking

GRASP with evolutionary path-relinking

Initialize pool as empty set

Repeat outer loop

Repeat inner loop

1) f ← construct greedy randomized
2) f ← local search(f)
3) If pool not empty: select f' from pool
3) f ← mixed path-relinking (f, f')
4) Attempt to update pool with f

pool ← evolutionary-PR(pool)

Mixed path-relinking

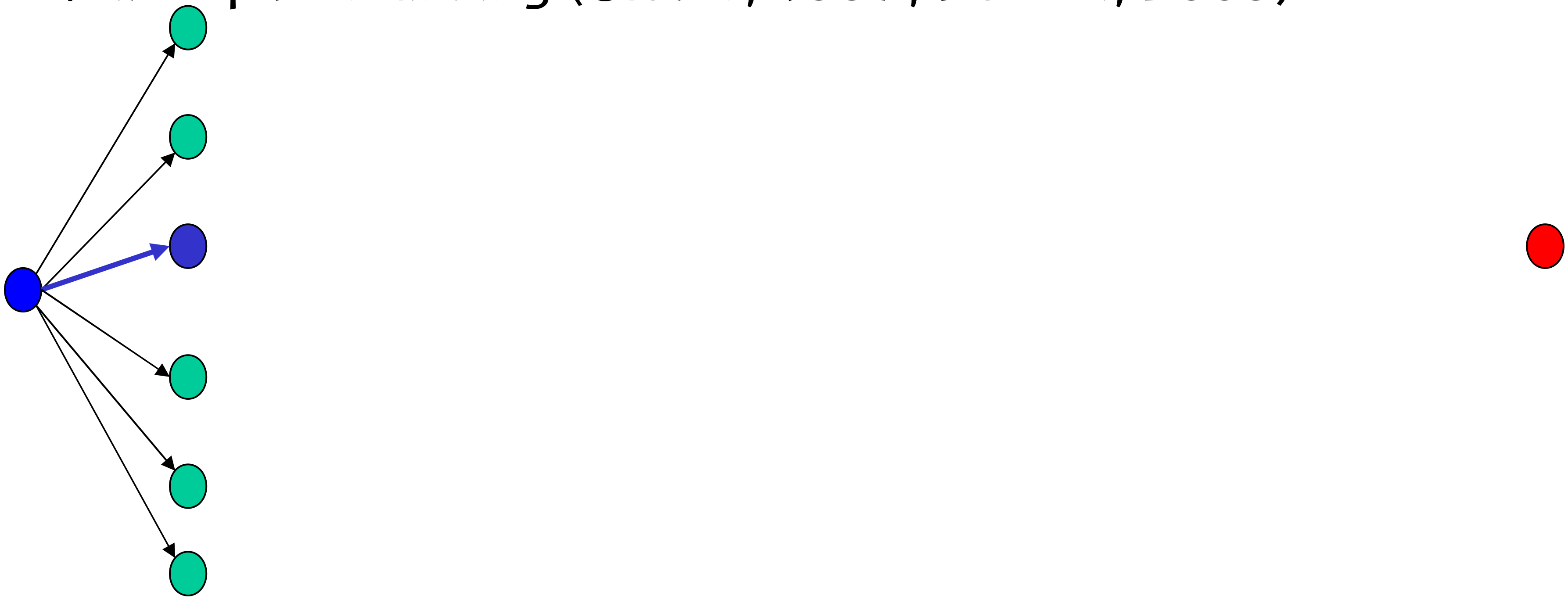
Variants: trade-offs between computation time and solution quality

Mixed path-relinking (Glover, 1997; Rosseti, 2003)

Mixed path-relinking

Variants: trade-offs between computation time and solution quality

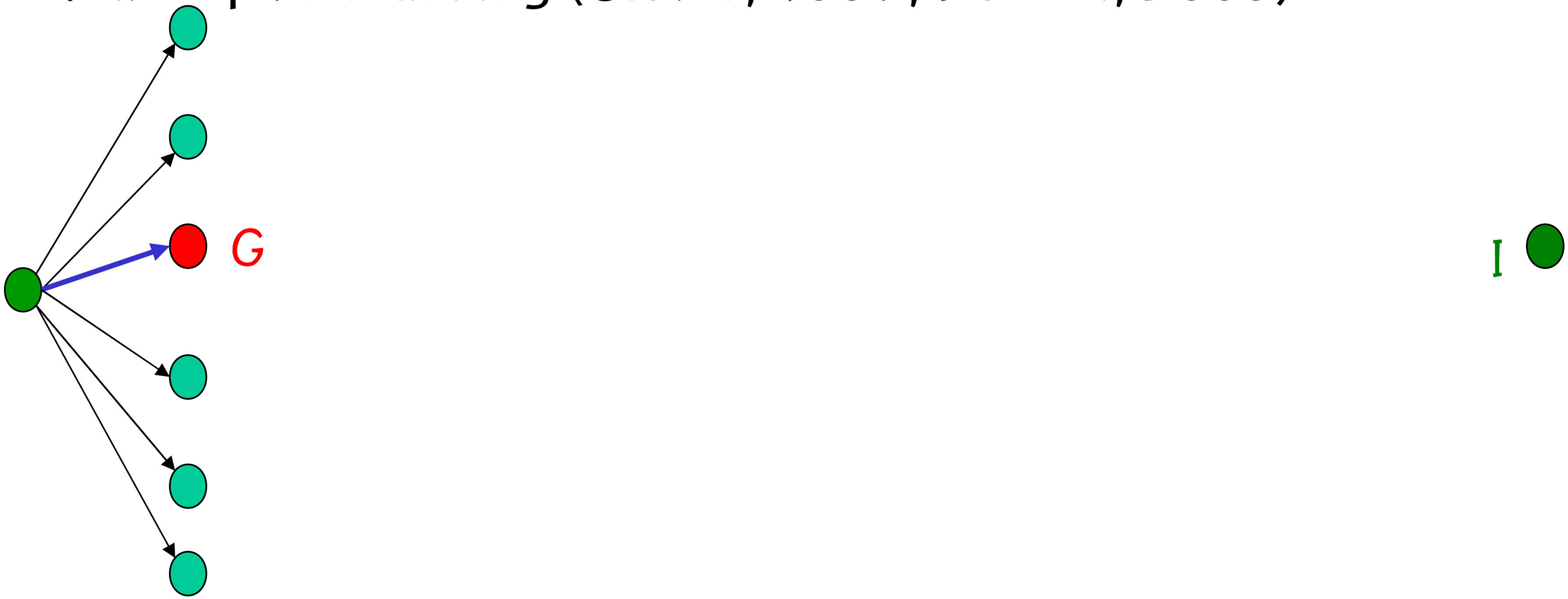
Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

Variants: trade-offs between computation time and solution quality

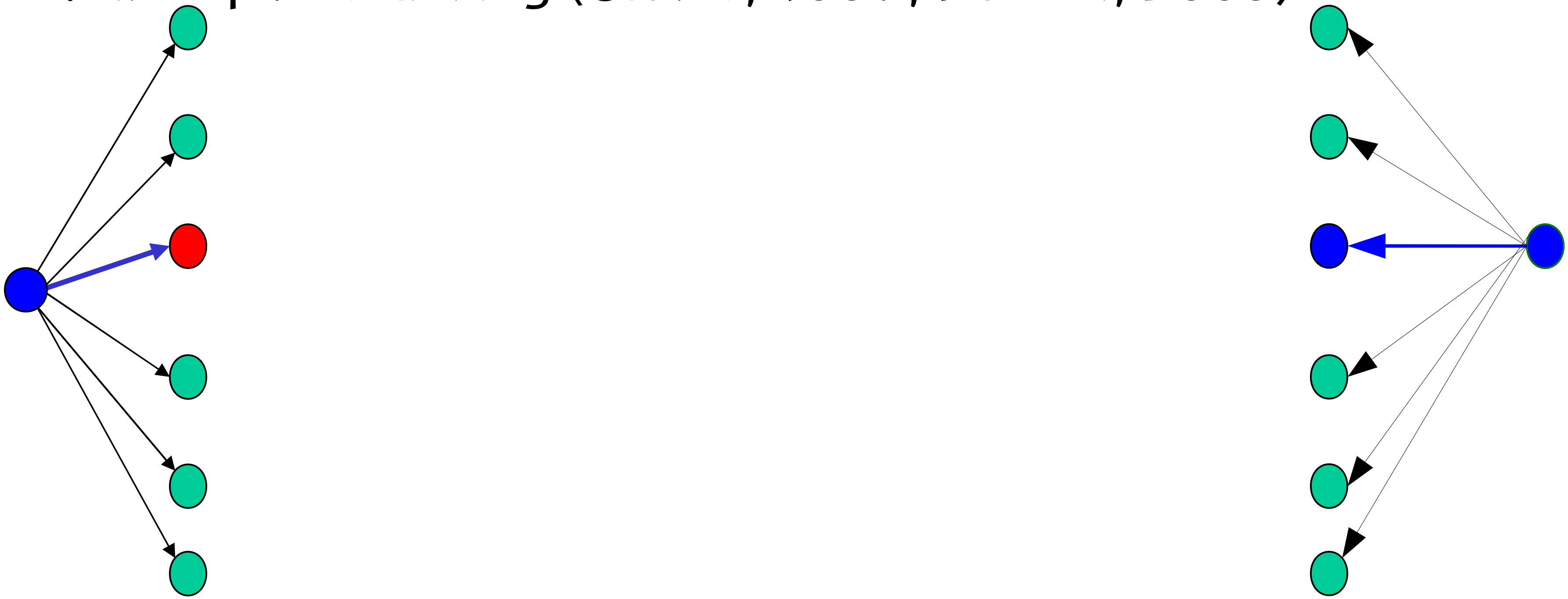
Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

Variants: trade-offs between computation time and solution quality

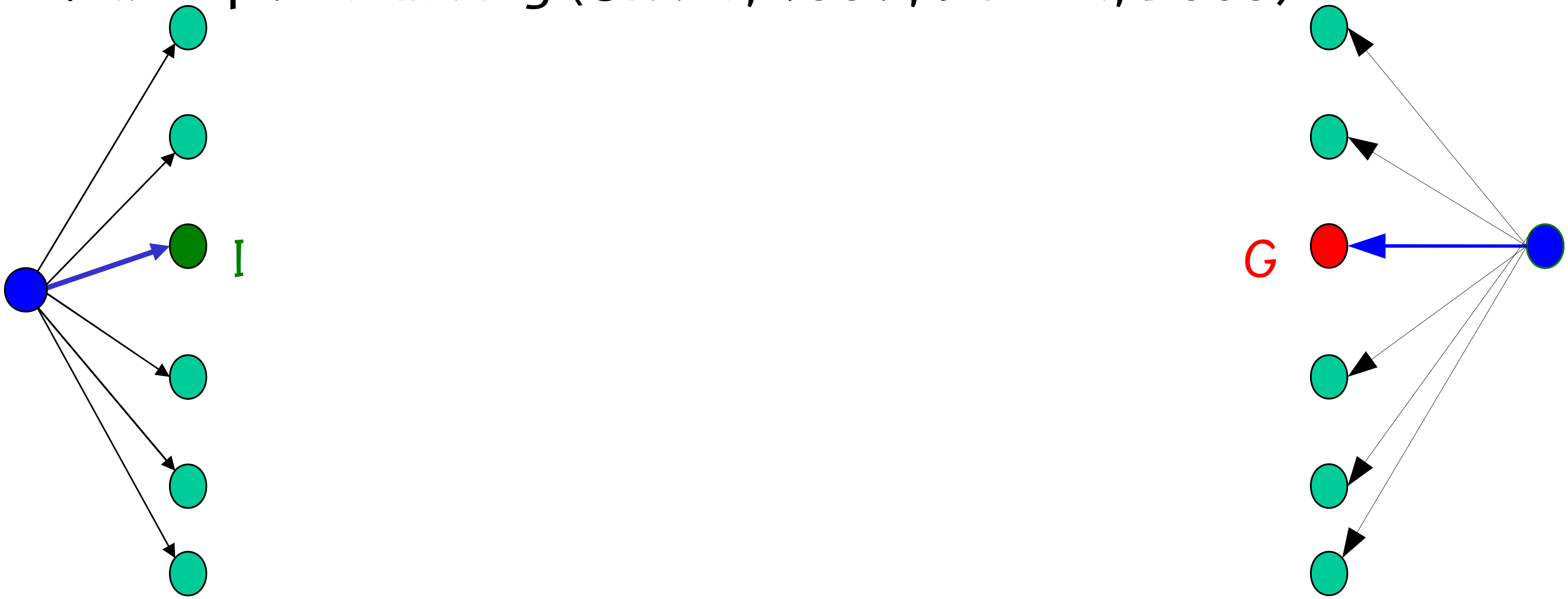
Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

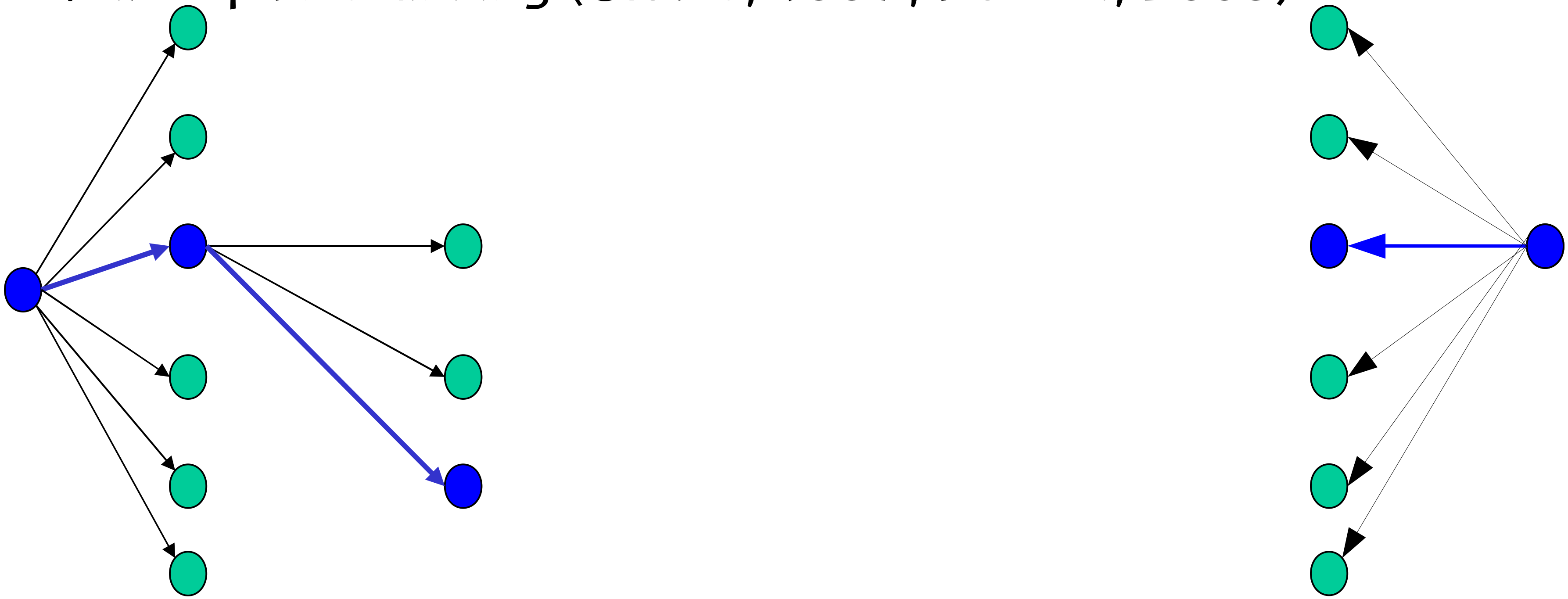
Variants: trade-offs between computation time and solution quality

Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

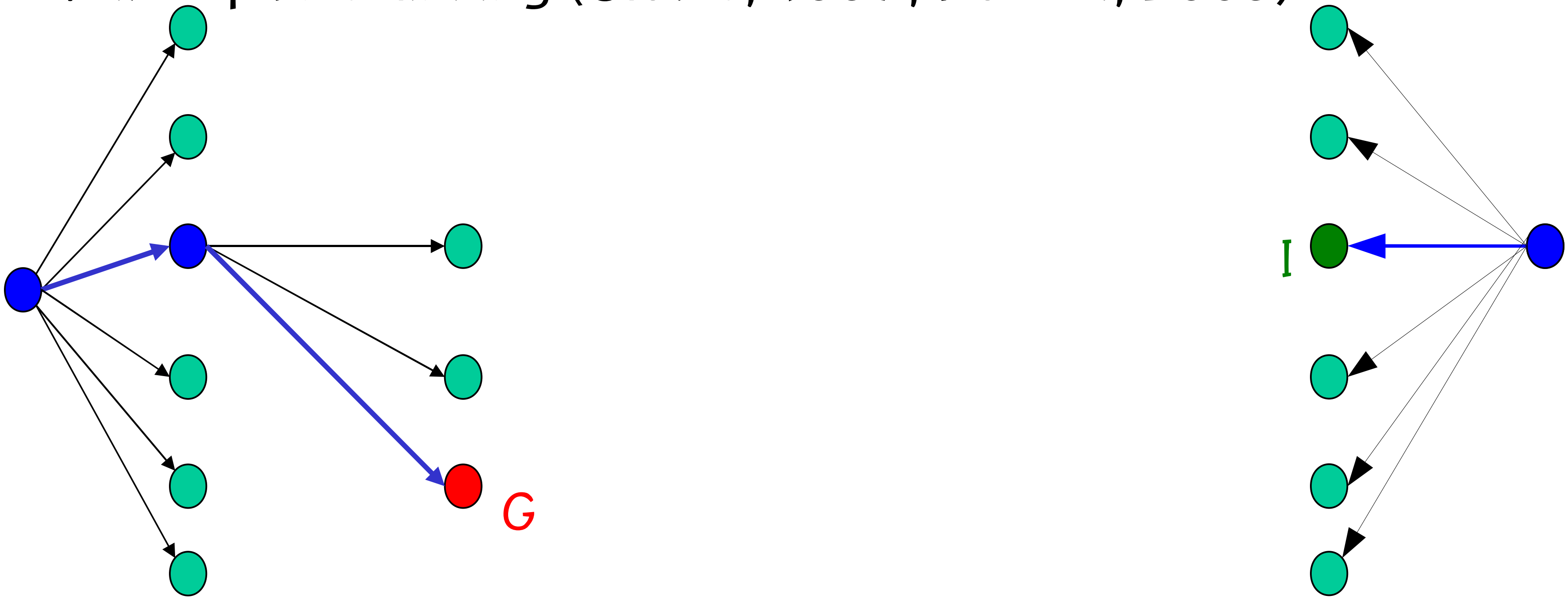
- Variants: trade-offs between computation time and solution quality
  - Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

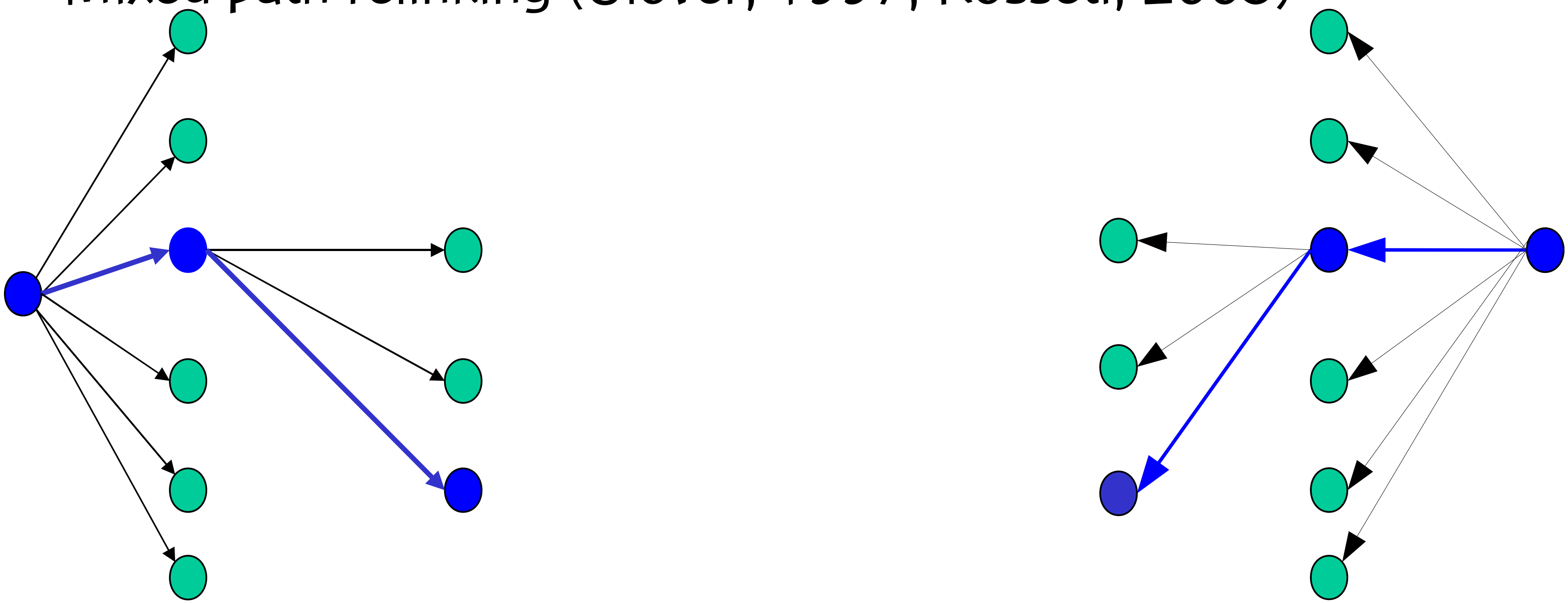
Variants: trade-offs between computation time and solution quality

Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

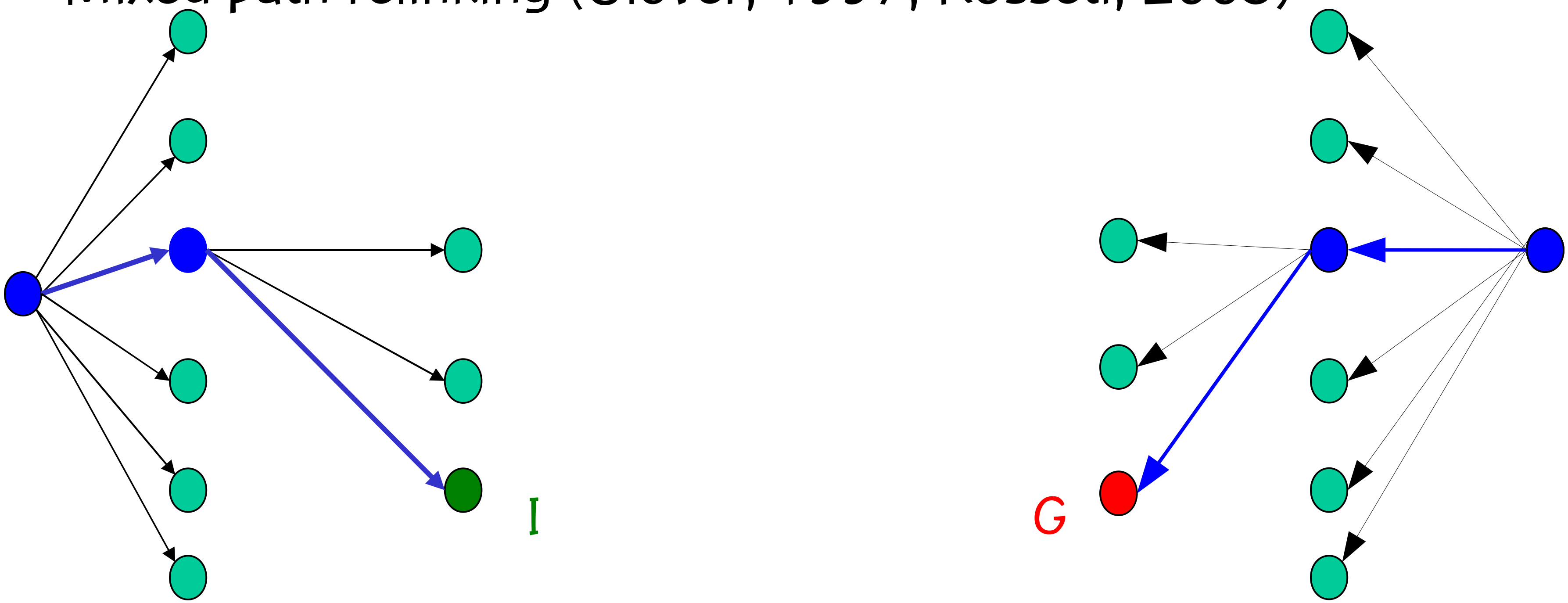
- Variants: trade-offs between computation time and solution quality
  - Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

Variants: trade-offs between computation time and solution quality

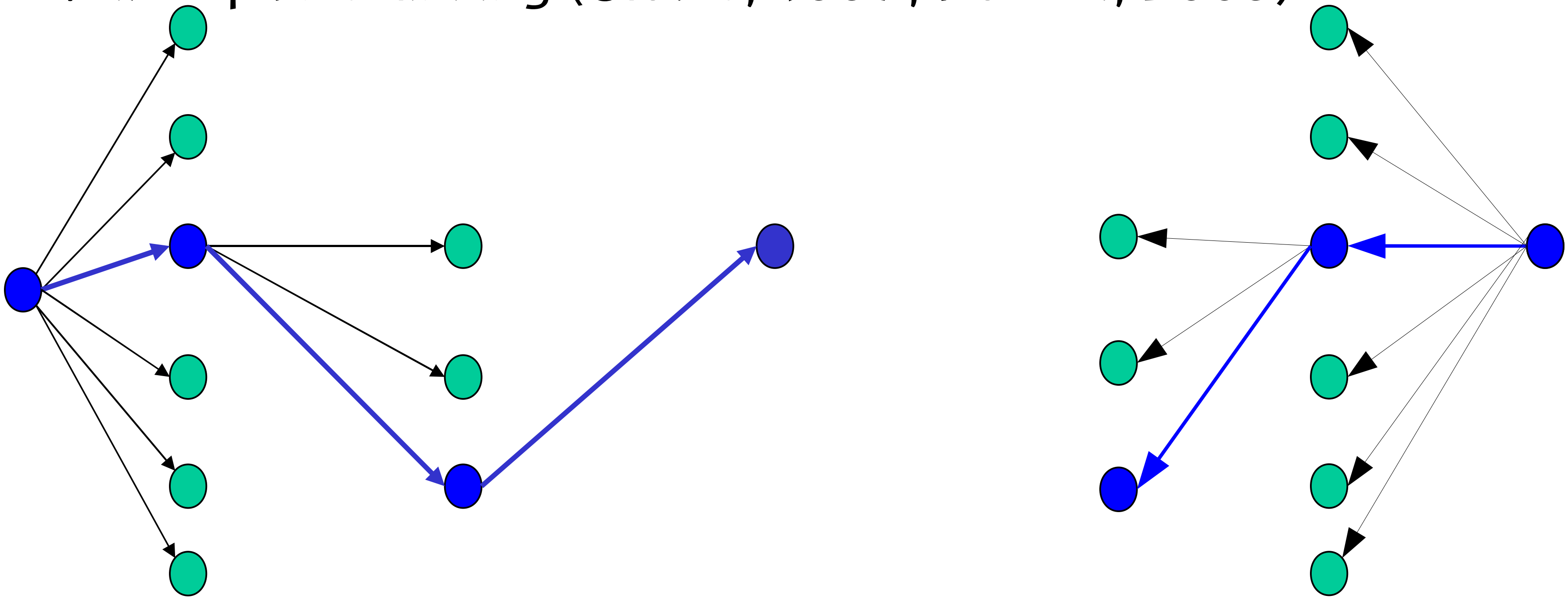
Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

Variants: trade-offs between computation time and solution quality

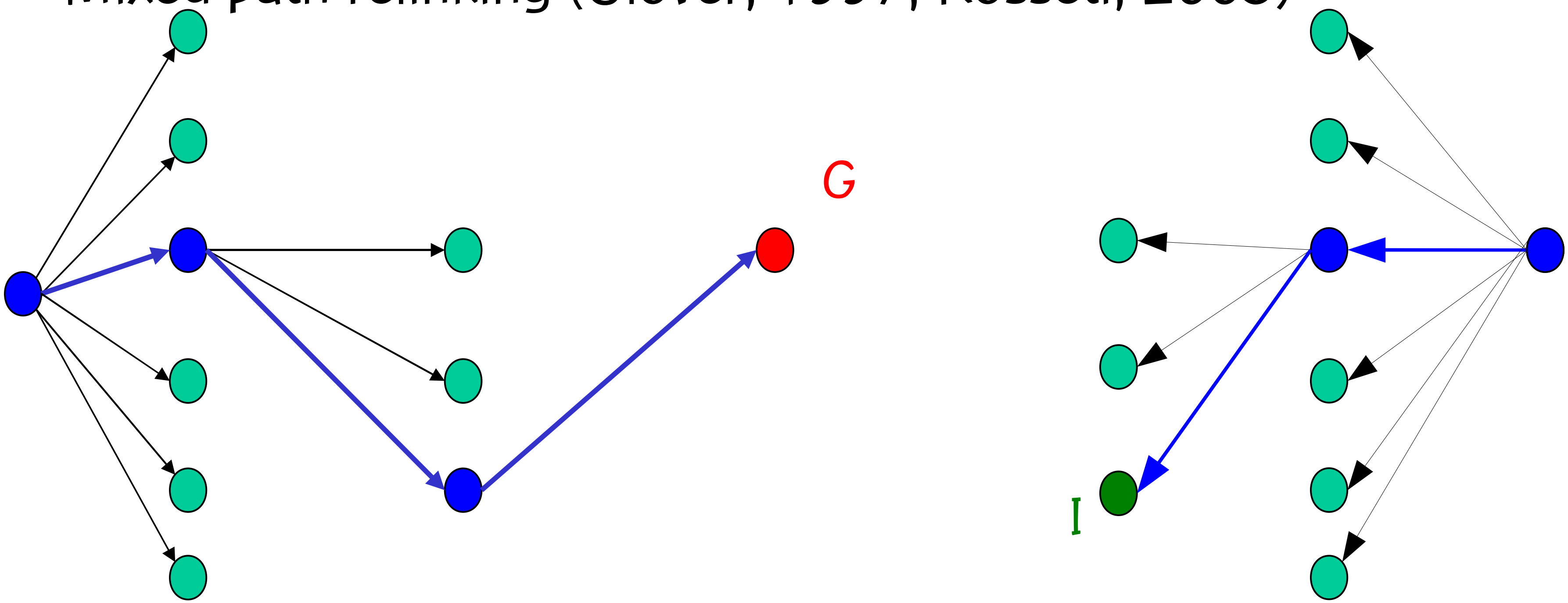
Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

Variants: trade-offs between computation time and solution quality

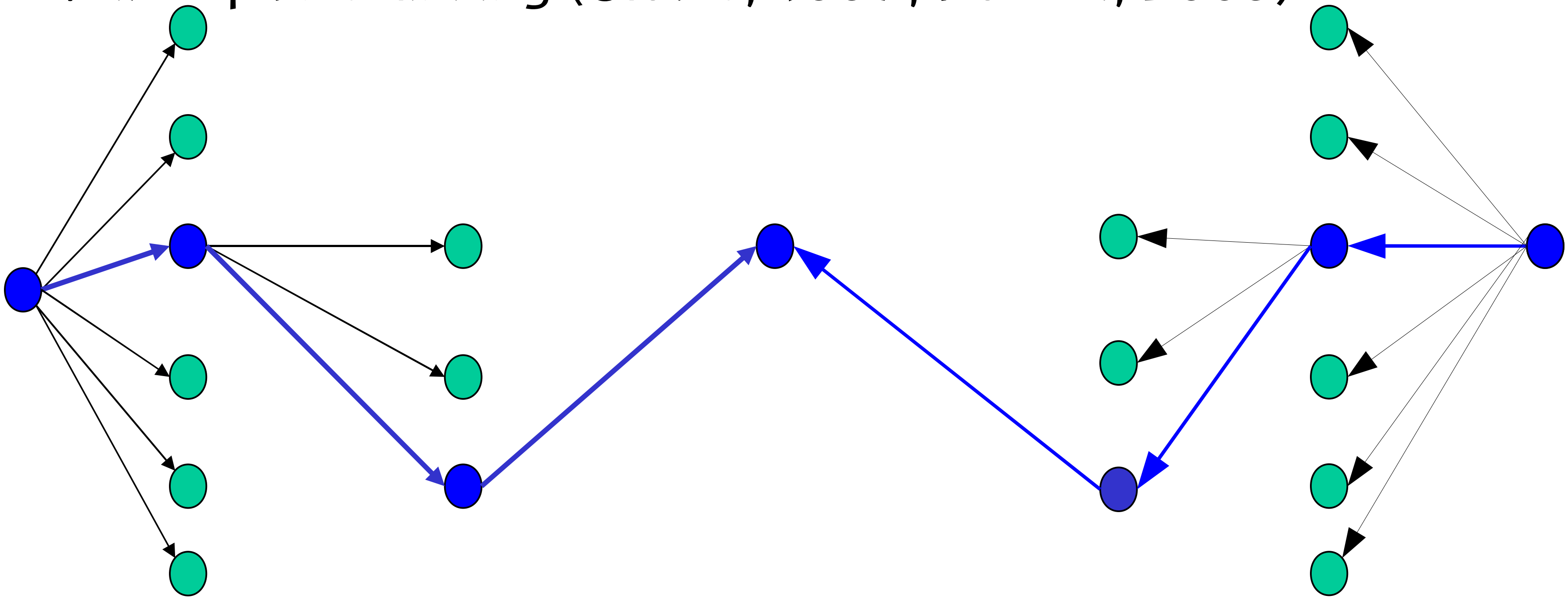
Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

Variants: trade-offs between computation time and solution quality

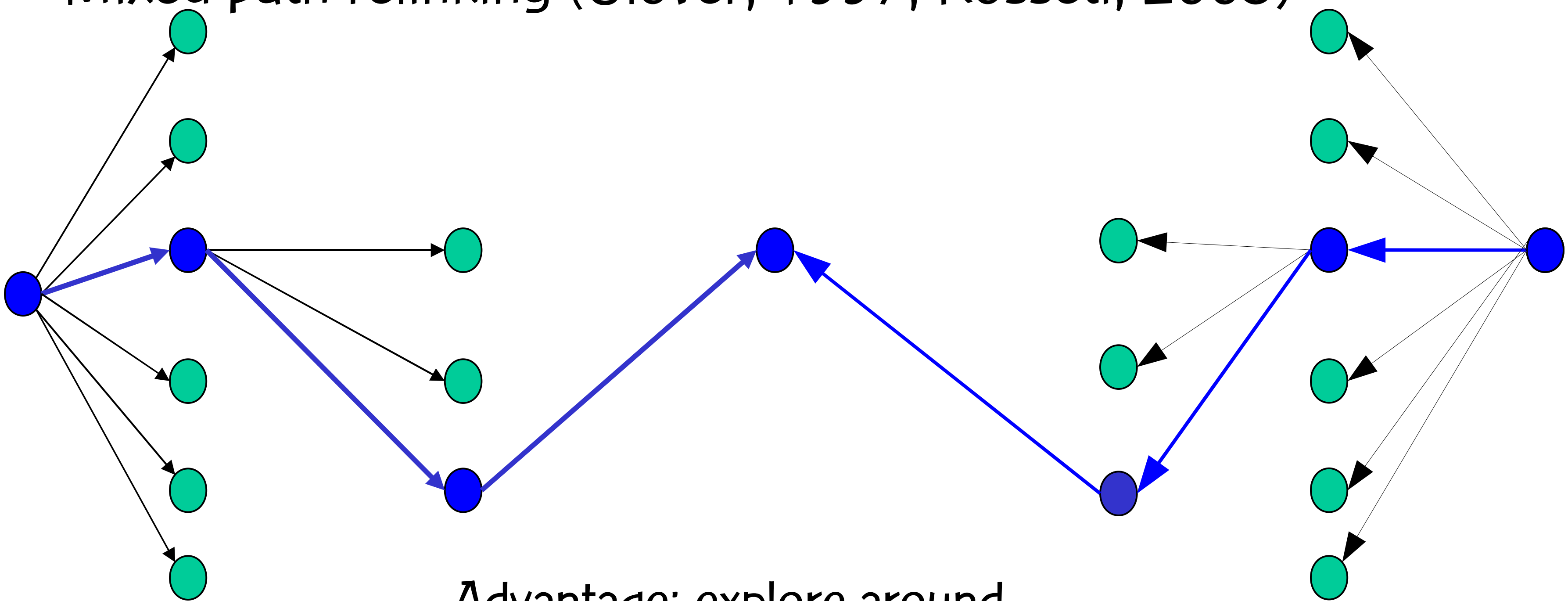
Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Mixed path-relinking

Variants: trade-offs between computation time and solution quality

Mixed path-relinking (Glover, 1997; Rosseti, 2003)



Advantage: explore around neighborhoods of both input solutions.

Pool management

- Pool has at most p (e.g. $p = 10$ elements) ordered from best {f(1)} to worst {f(p)}.
- Let $AB_{f(1)}(G)$ be the antibandwidth of the best labeling {f(1)} in the pool
- Labeling f is accepted to the pool if $AB_f(G) > AB_{f(1)}(G)$ or if $AB_f(G) > AB_{f(p)}(G)$ and $\Delta(f, \text{pool}) > \delta$, where

$$\Delta(f, \text{pool}) = \min\{\sum_{k=1}^{n} |f(k) - f^i(k)| : i \in \text{pool}\}$$

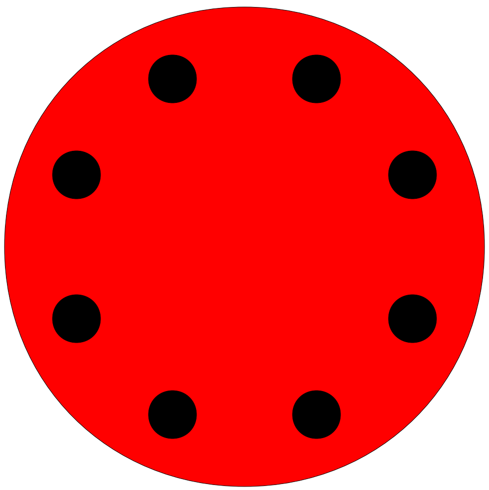
- If the pool is full and f is accepted into the pool: among all labelings f' such that $AB_{f'}(G) < AB_f(G)$ we remove from the pool the labeling closest to f.

Evolutionary path-relinking

(Resende & Werneck, 2004, 2006)

- Evolutionary path-relinking "evolves" the pool, i.e. transforms it into a pool of diverse elements whose solution values are better than those of the original pool.
- Evolutionary path-relinking can be used
  - as an intensification procedure at certain points of the solution process;
  - as a post-optimization procedure at the end of the solution process.

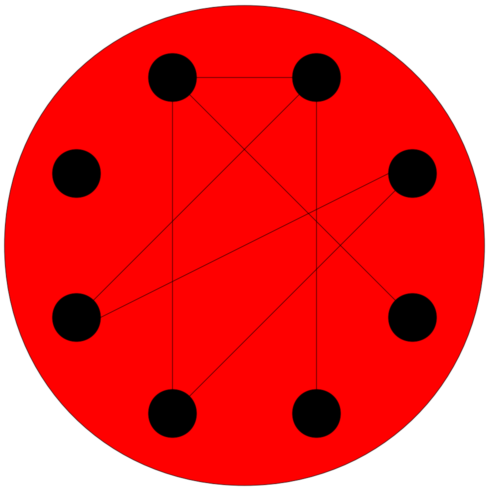
Evolutionary path-relinking (EvPR)



We use a variant of EvPR introduced in Resende, Martí, Gallego, & Duarte (2008)

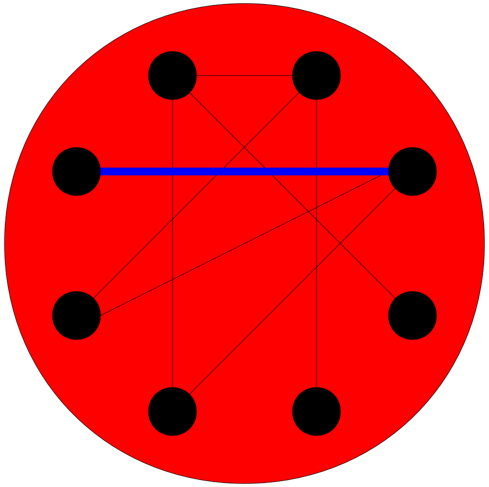
Start with the pool of elite solutions

Evolutionary path-relinking (EvPR)



While there exists a pair of pool solutions that have not yet been relinked:

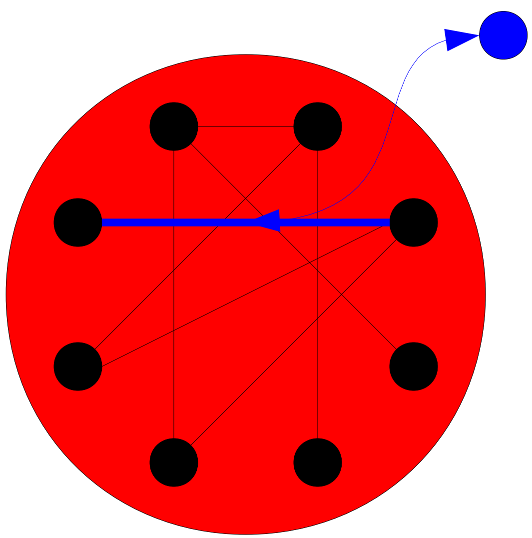
Evolutionary path-relinking (EvPR)



While there exists a pair of pool solutions that have not yet been relinked:

Apply mixed path-relinking between pair

Evolutionary path-relinking (EvPR)

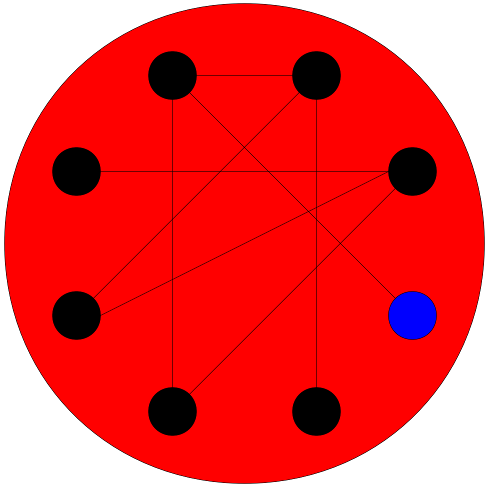


While there exists a pair of pool solutions that have not yet been relinked:

Apply mixed path-relinking between pair

Solution of path-relinking is candidate to enter the pool: if accepted, it replaces closest solution with smaller antibandwidth

Evolutionary path-relinking (EvPR)

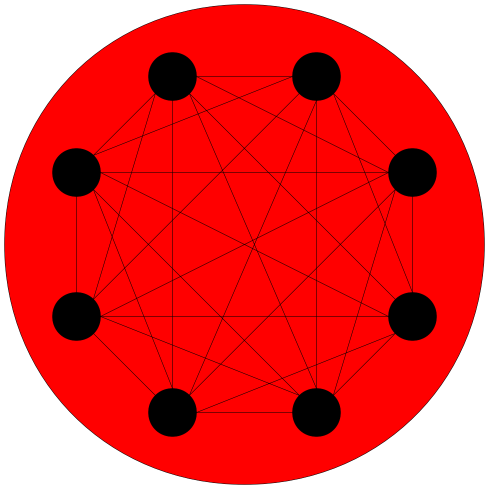


While there exists a pair of pool solutions that have not yet been relinked:

Apply mixed path-relinking between pair

Solution of path-relinking is candidate to enter the pool: if accepted, it replaces closest solution with smaller antibandwidth

Evolutionary path-relinking (EvPR)



EvPR ends when all pairs of pool solutions have been relinked and resulting labelings are not accepted to enter the pool.

Preliminary experimental results

Experiments

- Heuristics were coded in C and testing was done on a 3.0 GHz Pentium 4 PC with 3 Gb of memory
- CPLEX 11.1 was used to solve the integer program on a 1.6 GHz Itanium 2 computer with 256 Gb of memory
- Four sets of test problems serve as our benchmark

Experiments

- Test problems derived from the Harwell-Boeing Sparse Matrix Collection
  – 12 small instances (having between 30 and 100 vertices)
  – 12 large instances (having between 400 and 900 vertices)
- 2-dim meshes with optimal solutions known by construction (Raspaud et al., 2008)
  – 12 small instances (having between 90 and 120 vertices)
  – 12 large instances (having between 900 and 1200 vertices)

Experiments

All instances are available at http://www.uv.es/rmarti

- Test problems from the Harwell-Boeing Sparse Matrix Collection
  - 12 small instances (having between 30 and 100 vertices)
  - 12 large instances (having between 400 and 900 vertices)
- 2-dim meshes with optimal solutions known by construction (Raspaud et al., 2008)
  - 12 small instances (having between 90 and 120 vertices)
  - 12 large instances (having between 900 and 1200 vertices)

Integer programming: small Harwell-Boeing instances

name	n	m	nz	Iters million	B&B million	Time (secs)	Soln	UB
bcspwr01	209	1607	4931	74.8	3.7	14641	**17**	**17**
bcspwr02	265	2510	7675	426.6	8.9	>24h	21	22
ibm32	276	1147	3792	27.5	0.5	5709	**9**	**9**
pores1	296	1034	3524	352.6	16.9	>24h	6	8
curtis54	410	3095	9740	219.2	7.0	>24h	10	13
will57	425	3434	10763	216.0	3.8	>24h	12	14
bcsstk01	496	2529	8320	219.9	4.9	>24h	6	11
dwt234	675	13969	42363	91.9	1.7	>24h	23	58
ash85	693	7530	23427	116.2	3.8	>24h	12	27
bcspwr03	712	14222	43204	75.3	1.7	>24h	22	57
impcol.b	739	3822	12691	148.8	2.5	>24h	5	11
nos4	794	10348	31976	99.6	2.8	>24h	10	48

Integer programming: large Harwell-Boeing instances

name	n	m	nz thous	Iters million	B&B nodes	Time (secs)	Soln	UB
494bus	2654	245117	736	4.52	949	>24h	12	247
662bus	3798	439813	1321	1.35	408	>24h	16	331
685bus	4619	471193	1417	1.53	10	>24h	3	342
bcsstk06	8700	180541	559	5.97	406	>24h	1	210
bcsstk07	8700	180541	559	5.85	401	>24h	1	210
can445	4699	200153	608	3.35	321	>24h	1	221
can715	8095	514916	1557	1.89	16	>24h	1	357
dwt503	7033	256275	781	2.40	103	>24h	1	250
dwt592	6288	353313	1069	3.38	84	>24h	2	295
impcold	3809	182318	552	4.59	466	>24h	2	212
nos6	4605	457591	1377	2.37	48	>24h	4	337
sherman	4320	300004	905	3.16	107	>24h	5	272

Experiments with GRASP

- For each of the 48 instances, we apply G+evPR and G+PR 30 times
- G+evPR: 25 iterations of inner loop and 4 iterations of the outer loop (total of 100 GRASP iterations)
- G+PR: 250 iterations
- Size of elite set is 10

Deviation w.r.t. best or optimum

		minimum	maximum	average
Small grids	G+PR	2.9 %	5.9 %	3.8 %
	G+evPR	2.2 %	4.8 %	3.4 %
Large grids	G+PR	2.4 %	3.8 %	3.3 %
	G+evPR	2.2 %	3.6 %	3.0 %
Small H-B	G+PR	0.6 %	5.9 %	3.8 %
	G+evPR	0.0 %	5.9 %	3.1 %
Large H-B	G+PR	1.0 %	3.9 %	2.7 %
	G+evPR	0.0 %	3.4 %	2.1 %

CPU time (seconds)

		minimum	maximum	average
Small grids	G+PR	2.4	2.7	2.6
	G+evPR	4.0	5.4	4.7
Large grids	G+PR	1009.0	1081.6	1046.8
	G+evPR	2479.3	3281.1	2822.1
Small H-B	G+PR	1.0	1.1	1.1
	G+evPR	3.9	4.9	4.3
Large H-B	G+PR	194.3	200.9	197.6
	G+evPR	588.7	790.2	668.5

Number of best (optimal) solutions

		minimum	maximum	% best
Small grids	G+PR	0	30	23%
	G+evPR	0	30	25%
Large grids	G+PR	0	0	0%
	G+evPR	0	0	0%
Small H-B	G+PR	0	30	51%
	G+evPR	1	30	57%
Large H-B	G+PR	0	30	12%
	G+evPR	1	30	17%

Number of best (optimal) solutions

		minimum	maximum	% best
Small grids	G+PR	0	30	23%
	G+evPR	0	30	25%
Large grids	G+PR	0	0	0%
	G+evPR	0	0	0%
Small H-B	G+PR	0	30	51%
	G+evPR	1	30	57%
Large H-B	G+PR	0	30	12%
	G+evPR	1	30	17%

A minimum of 0 implies at least one instance (of the 12) for which all 30 runs failed to find the best/opt

Number of best (optimal) solutions

		minimum	maximum	% best
Small grids	G+PR	0	30	23%
	G+evPR	0	30	25%
Large grids	G+PR	0	0	0%
	G+evPR	0	0	0%
Small H-B	G+PR	0	30	51%
	G+evPR	1	30	57%
Large H-B	G+PR	0	30	12%
	G+evPR	1	30	17%

A maximum of 30 implies at least one instance (of the 12) for which all 30 runs found the best/opt

Number of best (optimal) solutions

		minimum	maximum	% best
Small grids	G+PR	0	30	23%
	G+evPR	0	30	25%
Large grids	G+PR	0	0	0%
	G+evPR	0	0	0%
Small H-B	G+PR	0	30	51%
	G+evPR	1	30	57%
Large H-B	G+PR	0	30	12%
	G+evPR	1	30	17%

%best = total number of runs that found best/opt / (12×30)

Small Harwell-Boeing instances (solution values)

name	IP CPLEX	G+PR			G+evPR		
		max	min	avg	max	min	avg
bcspwr01	**17**	17	16	16.13	17	16	16.40
bcspwr02	21	21	20	20.97	21	20	20.93
ibm32	**9**	9	8	8.30	9	8	8.27
pores1	6	6	6	6	6	6	6
curtis54	10	12	12	12	12	12	12
will57	12	13	12	12.3	13	12	12.43
bcsstk01	6	8	8	8	8	8	8
dwt234	23	51	49	49.5	51	49	49.67
ash85	12	21	19	19.87	22	19	20.30
bcspwr03	22	39	39	39	39	39	39
impcol.b	5	8	7	7.4	8	7	7.63
nos4	10	34	31	32.6	35	31	33.03

Large Harwell-Boeing instances (solution values)

name	IP CPLEX	G+PR			G+evPR		
		max	min	avg	max	min	avg
494bus	12	227	224	225.43	228	224	225.73
662bus	16	220	219	219.33	220	219	219.57
685bus	3	136	136	136.00	136	136	136.00
bcsstk06	1	32	31	31.2	33	31	31.57
bcsstk07	1	32	31	31.03	33	31	31.57
can445	1	82	75	78.2	85	78	80.67
can715	1	115	112	113.73	127	115	115.97
dwr503	1	53	51	51.97	58	51	53.73
dwr592	2	108	99	103.03	112	102	106.10
impcol.d	2	104	100	102.03	105	101	102.90
nos6	4	326	324	325.4	328	325	326.47
sherman	5	261	260	260.1	261	260	261.1

Concluding remarks

- We described a GRASP with evolutionary path-relinking for the antibandwidth problem.
- The antibandwidth problem has an important application in frequency assignment in cellular telephony.
- To complete the experiments, we will derive run time distributions for the heuristics. Preliminary results indicate that G+PR and G+evPR have similar run time distributions.
- We will also conclude the CPLEX runs on the mesh instances. Preliminary results indicate that CPLEX cannot solve optimally even the smallest of the mesh instances.
- Our current G+evPR implementation can be made more efficient, resulting in a reduction in the number of path-relinking operations in the evolutionary path-relinking procedure.

Coauthors

Rafael Martí & Abraham Duarte

Ricardo M. A. Silva

The End

These slides and a technical report
can be downloaded from my homepage:
http://www.research.att.com/~mgcr