Parameterized Maximum Path Coloring

Michael Lampis

September 9, 2011

Path Coloring Definition

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems

Path Coloring

Input: A graph G and a multi-set of paths on that graph
Constraint: Assign colors from $\{1, \ldots, W\}$ to the paths so that paths that share an edge receive different colors.
Objective: min W

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems

Path Coloring

Input: A graph G and a multi-set of paths on that graph
Constraint: Assign colors from $\{1, \ldots, W\}$ to the paths so that paths that share an edge receive different colors.
Objective: min W

- Graph could be undirected or bi-directed
- Instead of paths we could be given endpoints (Routing and Path Coloring)

- ❖ Path Coloring
- ❖ Example
- ❖ Known results
- ❖ Edge slicing
- ❖ Max PC
- ❖ Max PC hardness results
- ❖ DNP
- ❖ Reduction
- ❖ Reduction
- ❖ Complexity jump
- ❖ $(p\Delta, pW, pT)$-MaxPC
- ❖ (pT)-MaxPC binary trees
- ❖ (pT)-MaxPC binary trees
- ❖ Algoritm cont'd
- ❖ Open problems

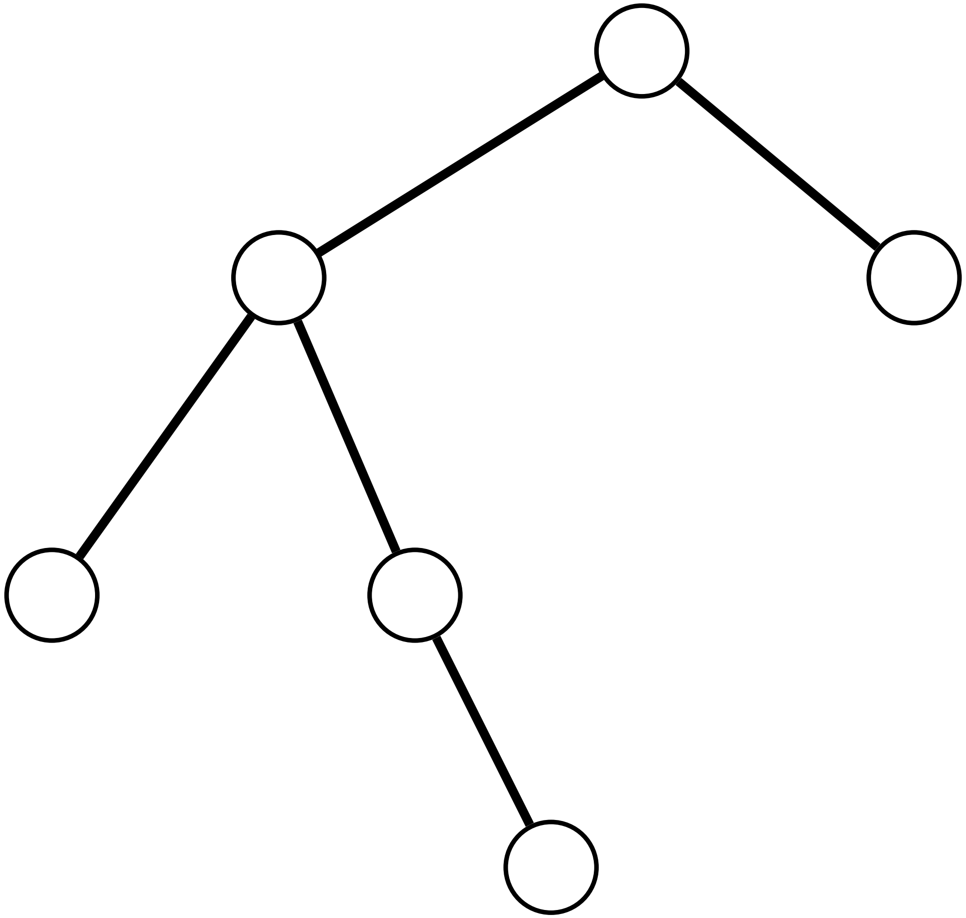
Path Coloring

Input: A graph G and a multi-set of paths on that graph
Constraint: Assign colors from $\{1, \ldots, W\}$ to the paths so that paths that share an edge receive different colors.
Objective: min W

- Graph could be undirected or bi-directed
- Instead of paths we could be given endpoints (Routing and Path Coloring)
- We'll mostly talk about trees ($\rightarrow$ unique routing)

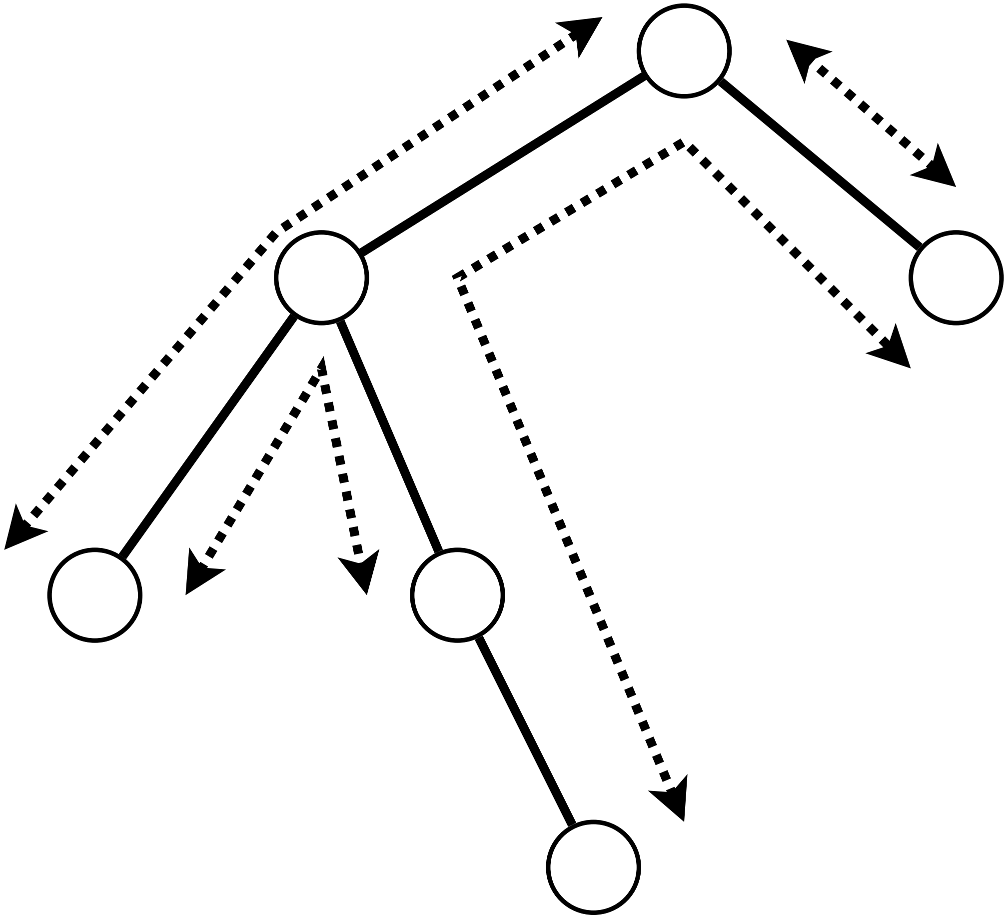
Example

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems



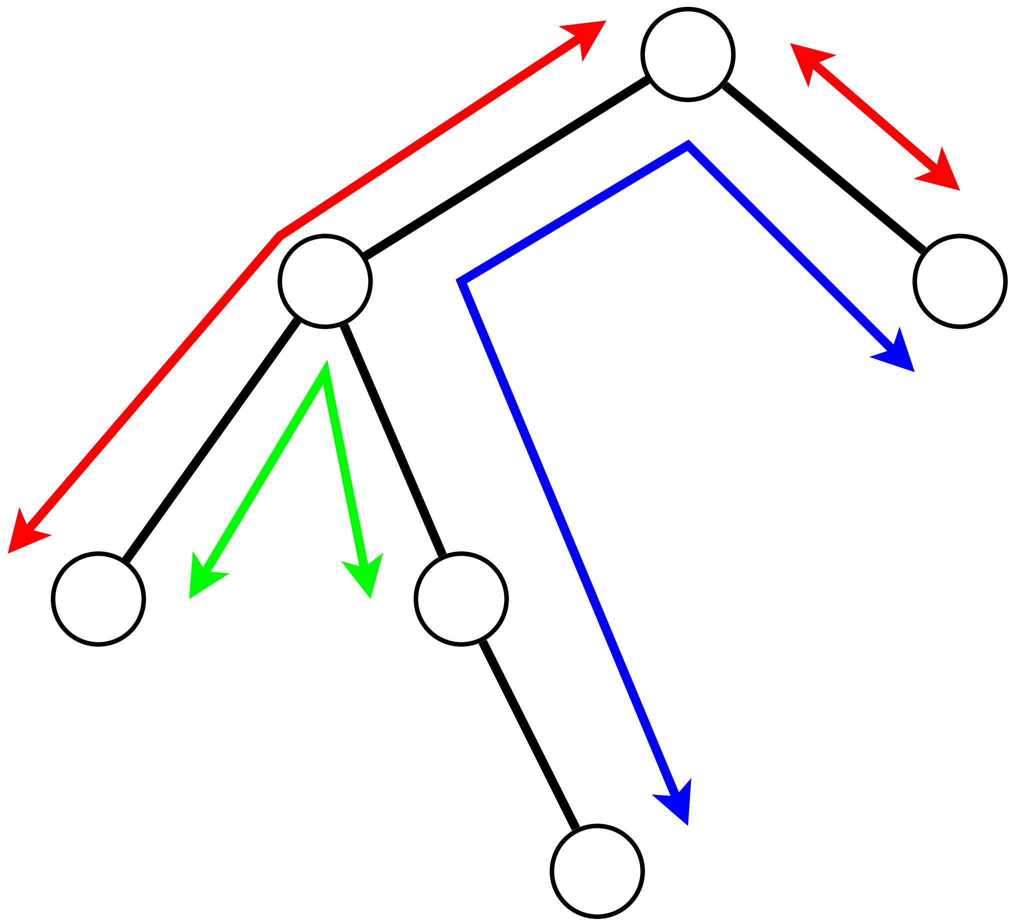
Example

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems



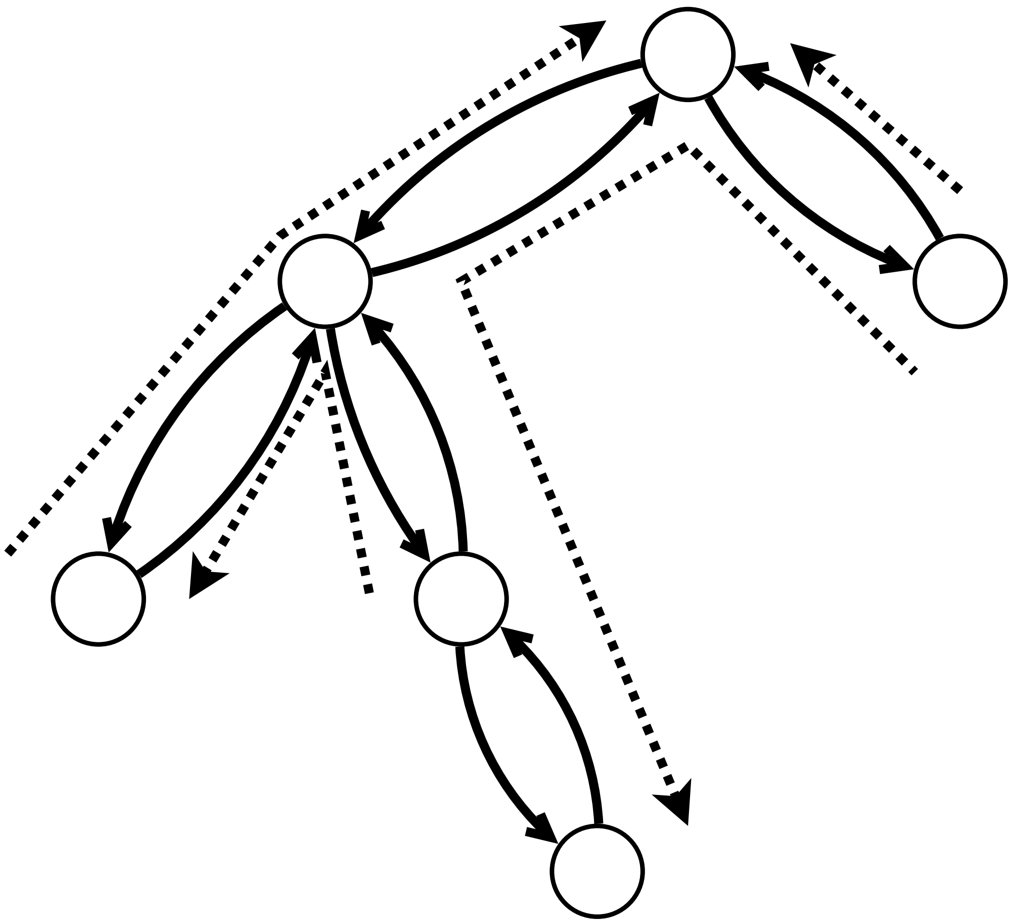
Example

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems



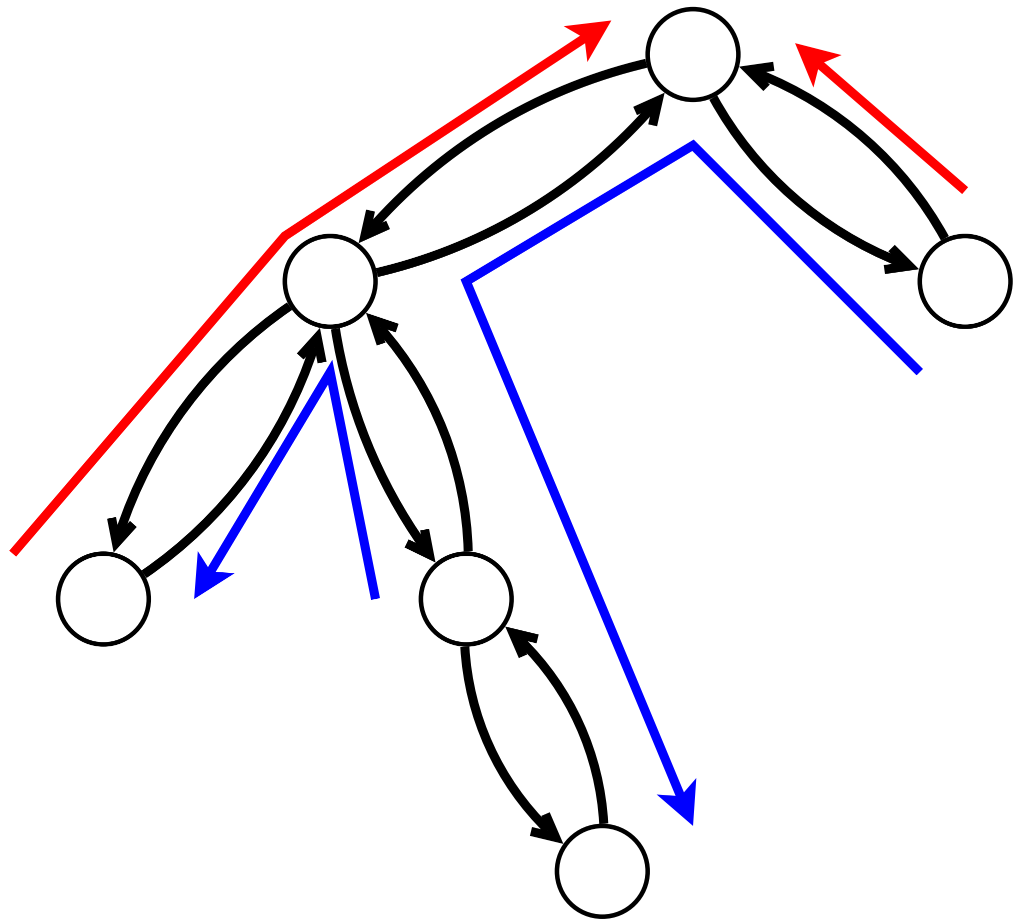
Example

- ❖ Path Coloring
- ❖ Example
- ❖ Known results
- ❖ Edge slicing
- ❖ Max PC
- ❖ Max PC hardness results
- ❖ DNP
- ❖ Reduction
- ❖ Reduction
- ❖ Complexity jump
- ❖ $(p\Delta, pW, pT)$-MaxPC
- ❖ (pT)-MaxPC binary trees
- ❖ (pT)-MaxPC binary trees
- ❖ Algoritm cont'd
- ❖ Open problems



Example

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems



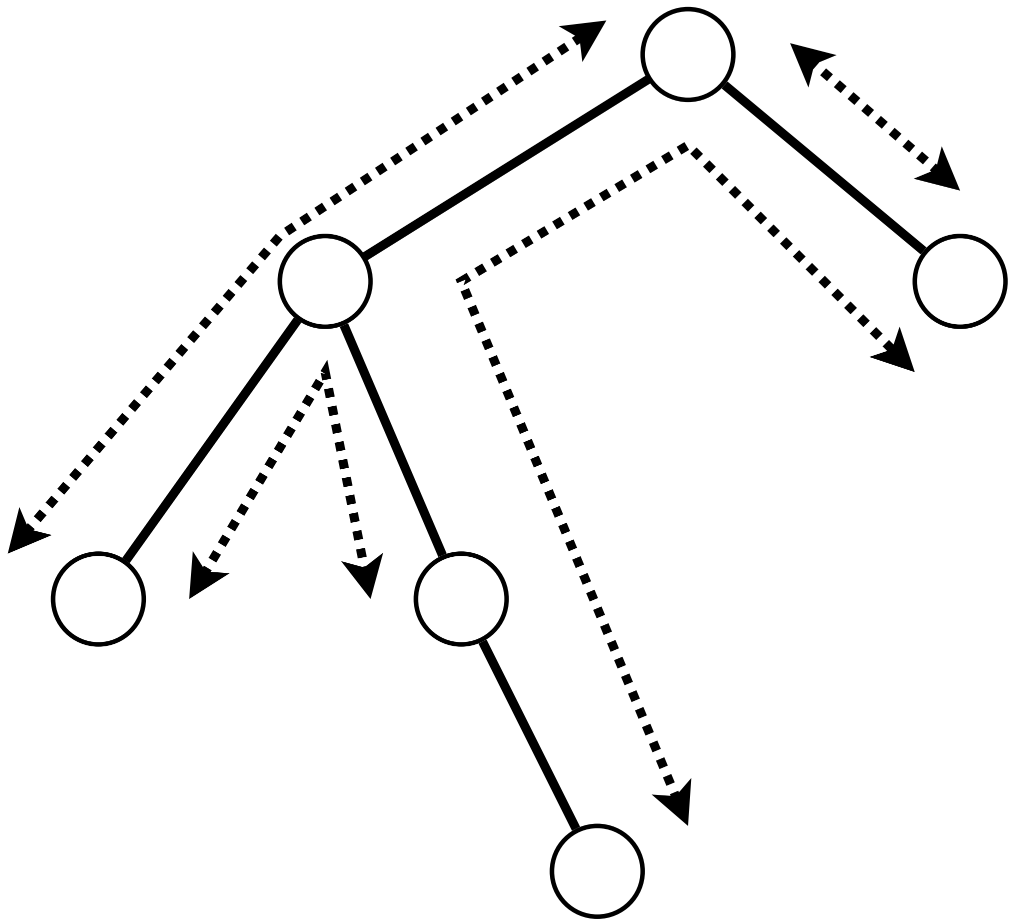
Known results

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

- PC is very hard!
  - NP-hard on stars [Erlebach, Jansen 2001]
  - NP-hard on rings [Garey, Johnson, Miller, Papadimitriou 1980]
  - NP-hard on bi-directed binary trees [Kumar, Panigrahy, Russel, Sundaram 1997]
- Good news: Thanks to a simple trick undirected trees are no harder than stars.

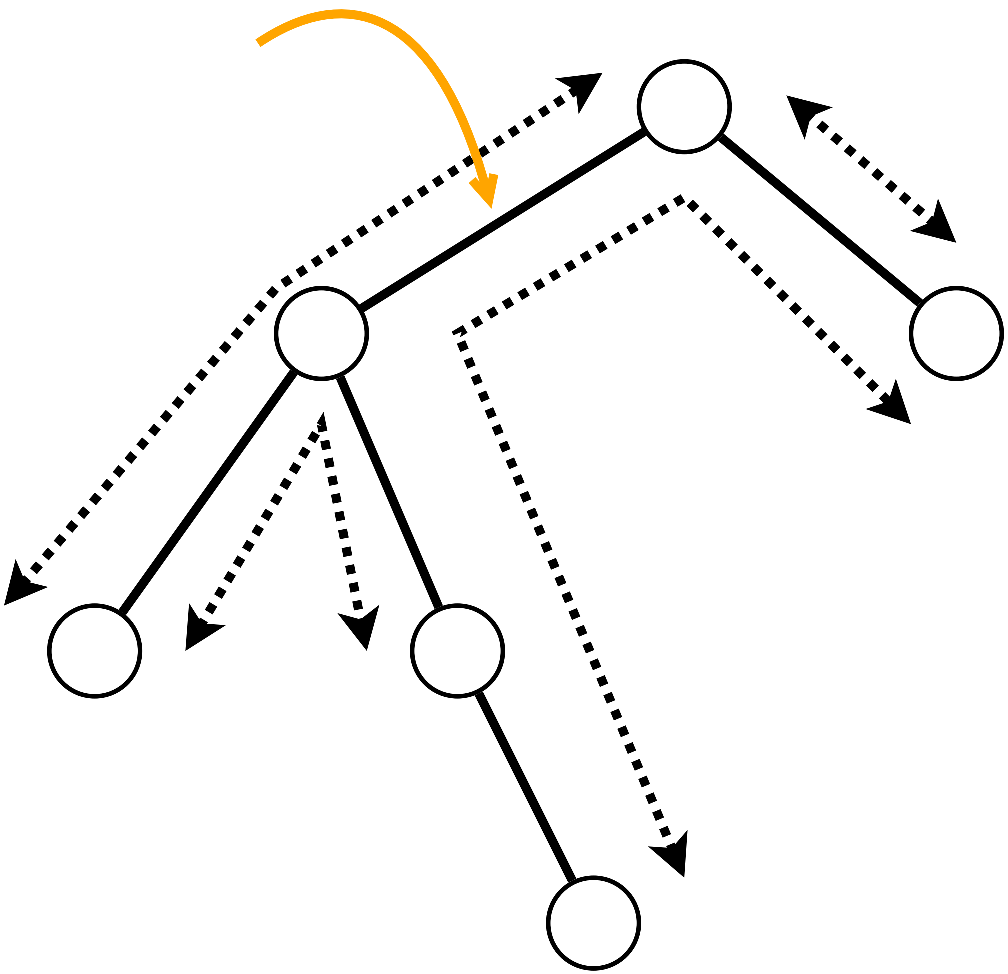
Edge slicing

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems



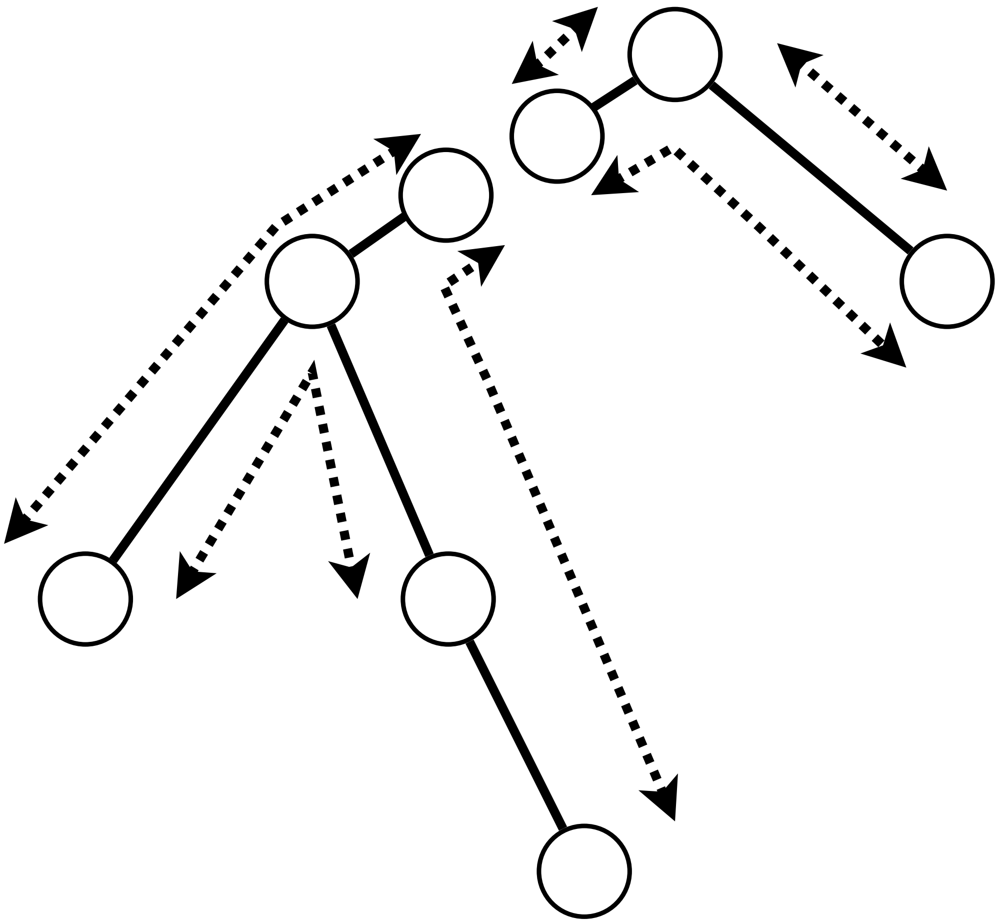
Edge slicing

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems



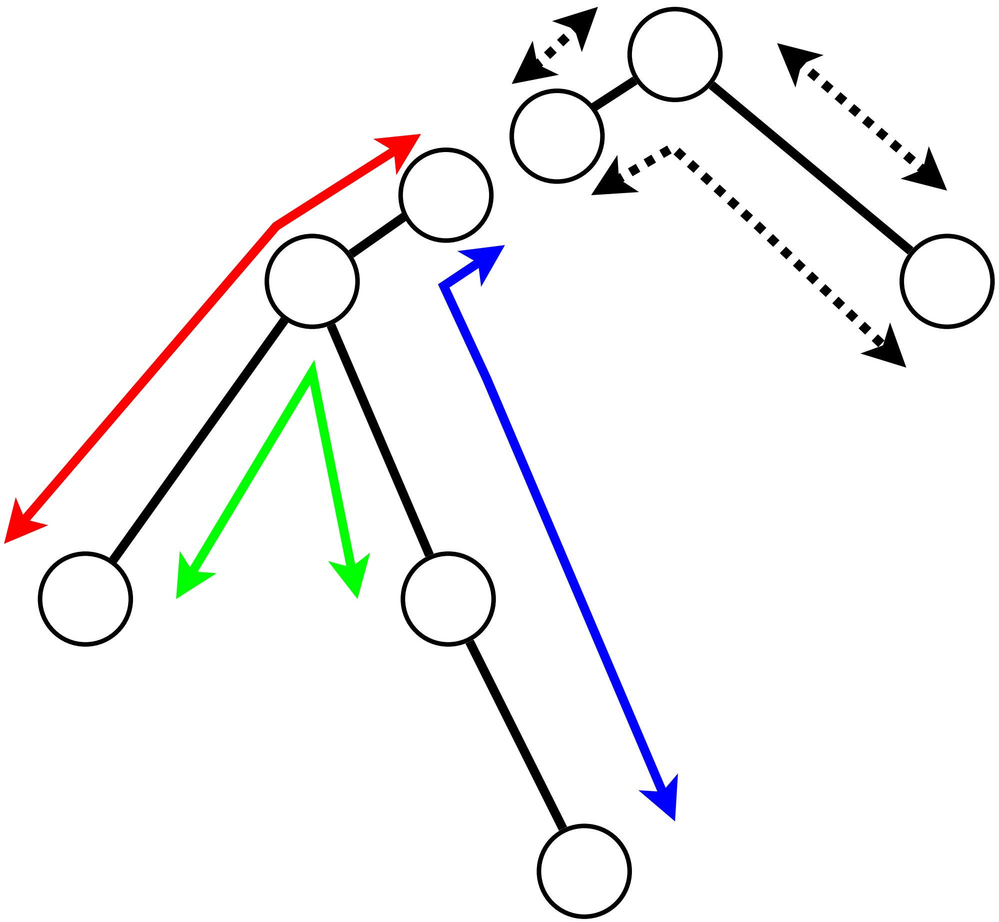
Edge slicing

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems



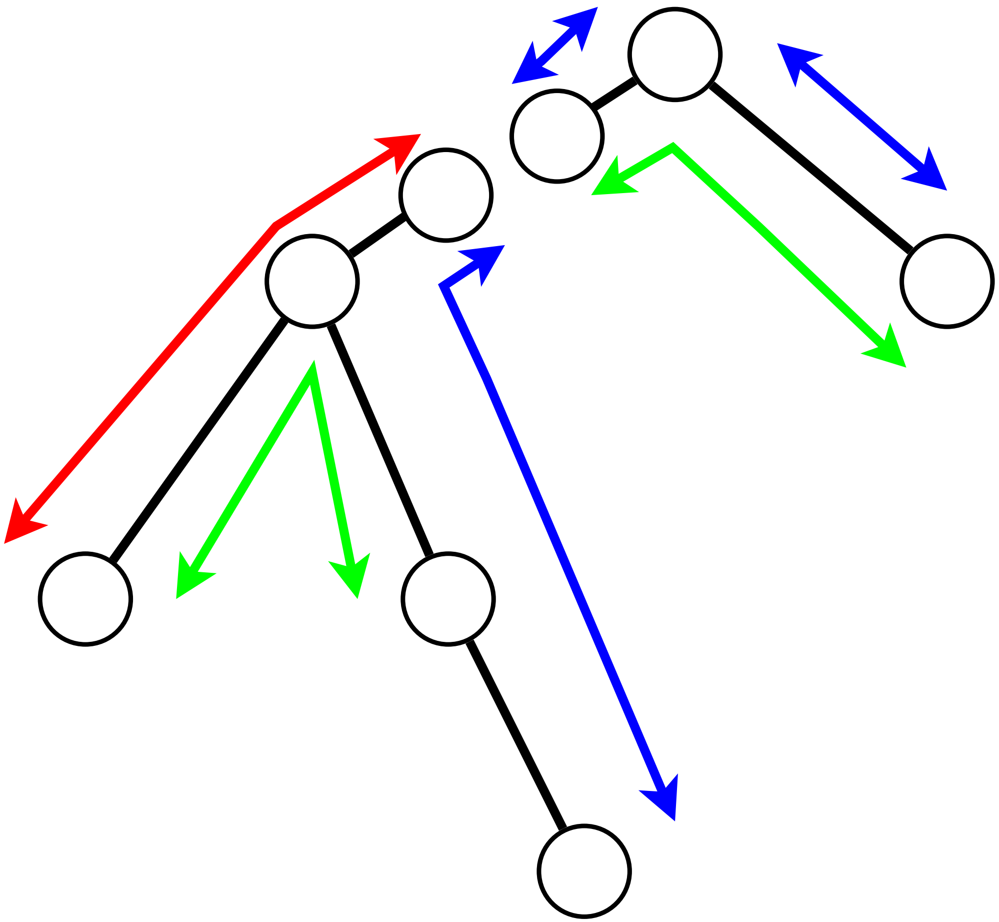
Edge slicing

- ❖ Path Coloring
- ❖ Example
- ❖ Known results
- ❖ Edge slicing
- ❖ Max PC
- ❖ Max PC hardness results
- ❖ DNP
- ❖ Reduction
- ❖ Reduction
- ❖ Complexity jump
- ❖ $(p\Delta, pW, pT)$-MaxPC
- ❖ (pT)-MaxPC binary trees
- ❖ (pT)-MaxPC binary trees
- ❖ Algoritm cont'd
- ❖ Open problems



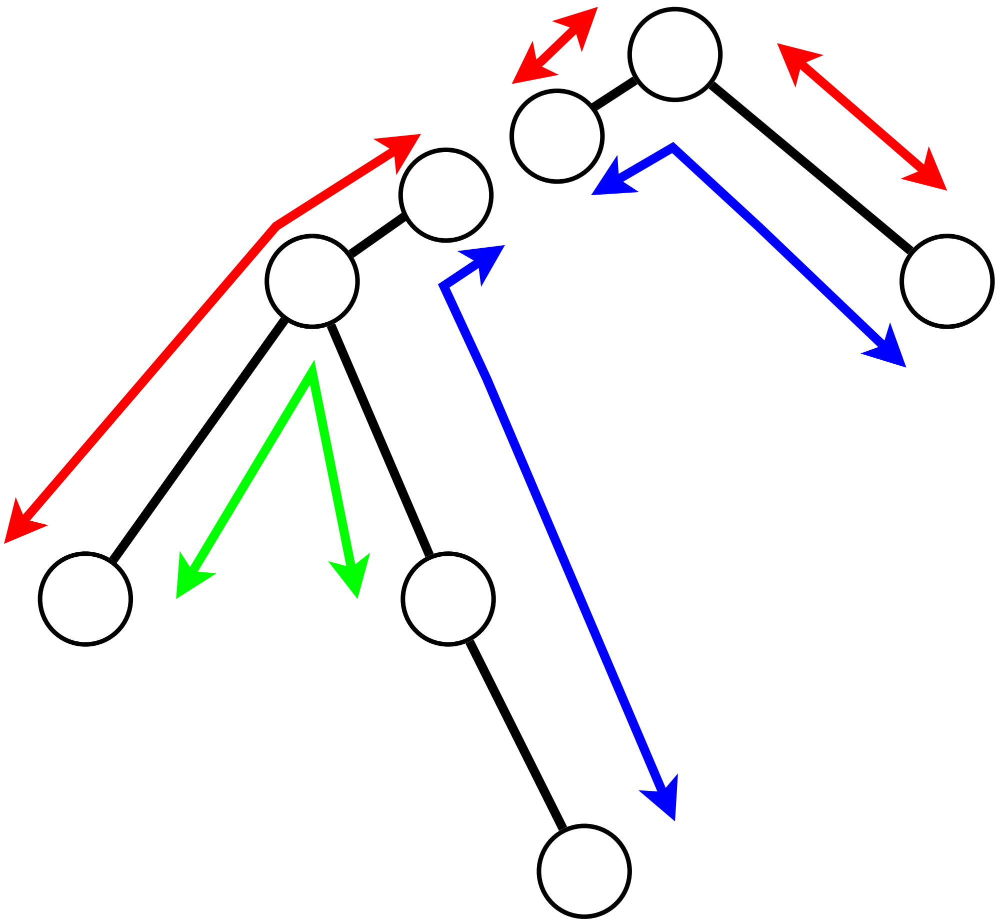
Edge slicing

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems



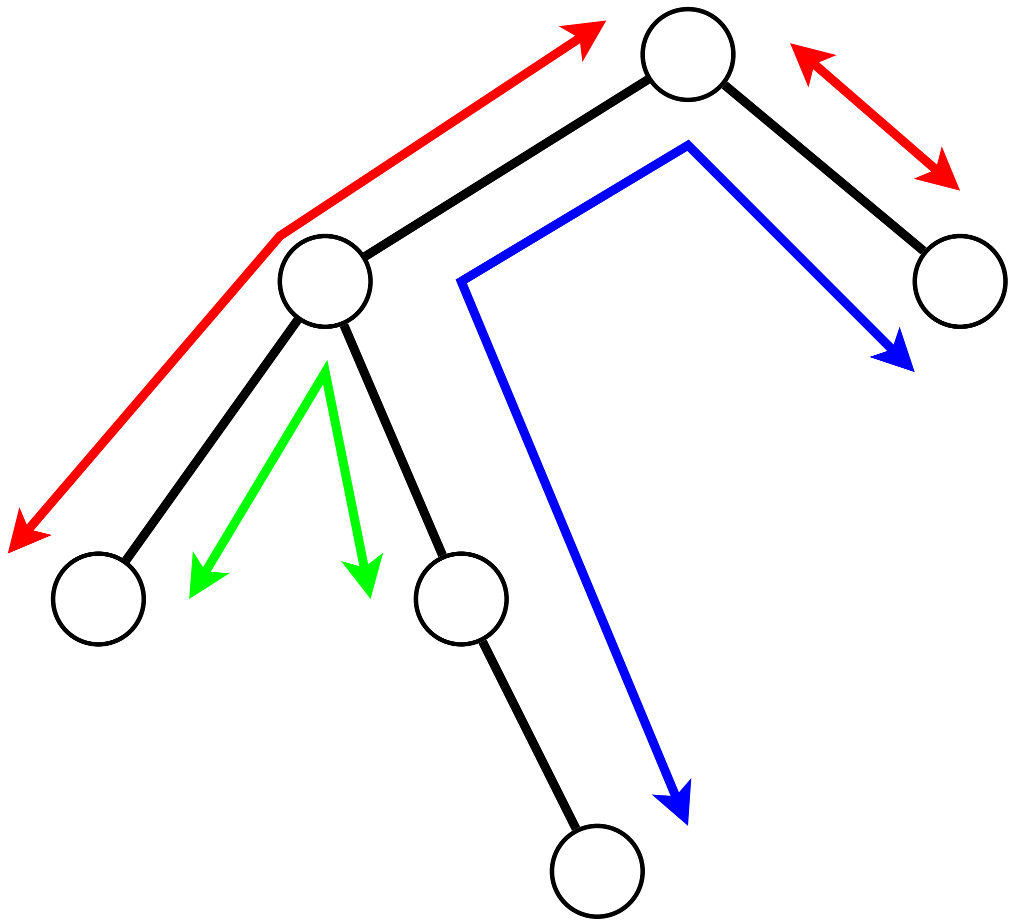
Edge slicing

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems



Edge slicing

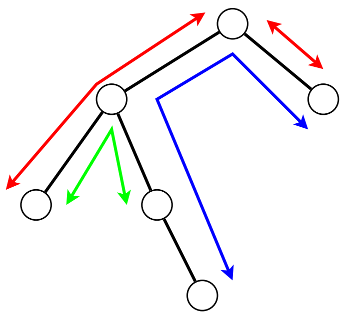
❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems



Edge slicing

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

- Repeated edge slicing can break down any undirected tree to a star
- If we could solve PC on stars $\rightarrow$ poly-time algorithm (we can't!)
  - But FPT algorithm when parameterized by Δ. [Erlebach, Jansen 2001]
- Ironically, this doesn't work for bi-directed trees, where stars are easy.
  - But FPT algorithm when parameterized by $\Delta + W$. [Erlebach, Jansen 2001]



❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems

Max Path Coloring

Input: A graph G and a multi-set of paths on that graph, color buget W
Constraint: Assign colors from $\{1, \ldots, W\}$ to B of the paths so that paths that share an edge receive different colors.
Objective: max B

- Strict generalization of PC as a decision problem
  - $\rightarrow$ At least as hard to solve exactly

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems

Max Path Coloring

Input: A graph G and a multi-set of paths on that graph, color buget W
Constraint: Assign colors from $\{1, \ldots, W\}$ to B of the paths so that paths that share an edge receive different colors.
Objective: max B

- Strict generalization of PC as a decision problem
  - $\rightarrow$ At least as hard to solve exactly
- Max PC is solvable in $n^{\Delta W}$ on trees. [Erlebach, Jansen 1998]
- Can we do this in FPT time for either parameter?

Max PC hardness results

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

- An $n^{\Delta W}$ algorithm is known to solve Max PC exactly on trees. Can we do better?

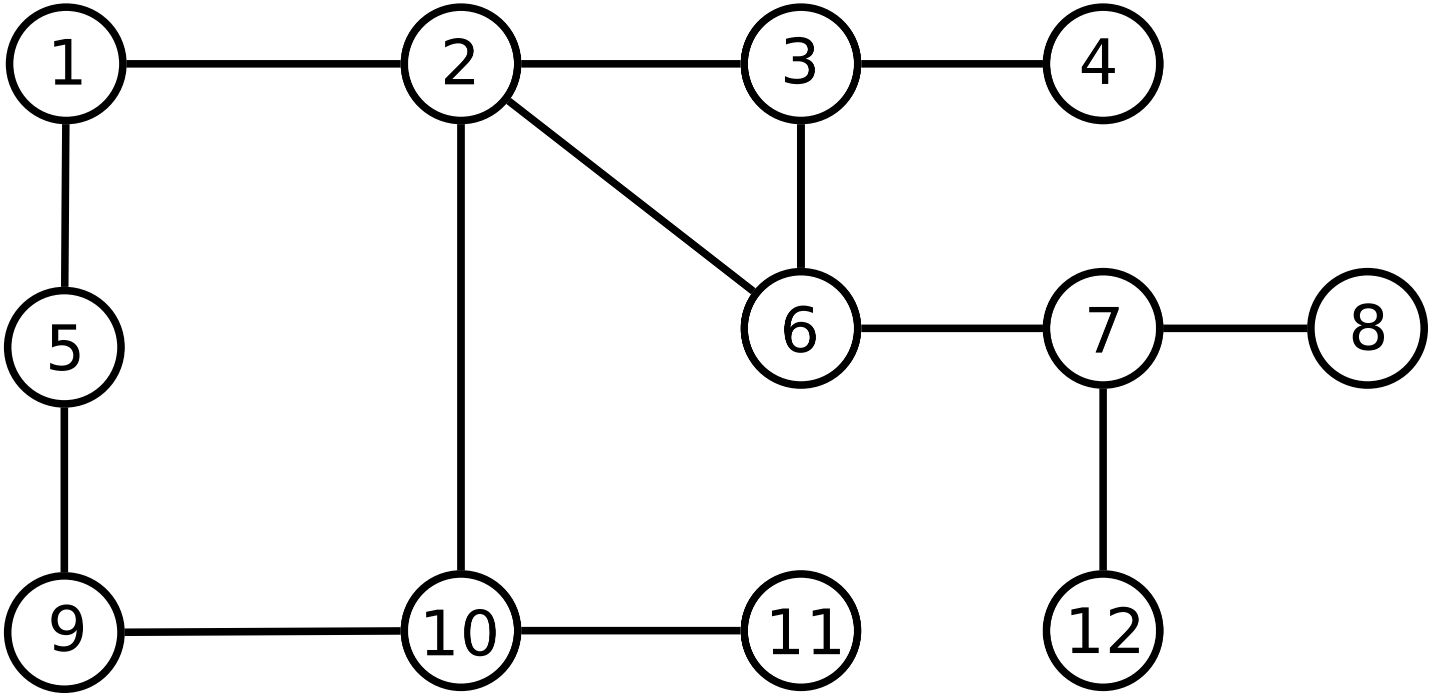
Max PC hardness results

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

- An $n^{\Delta W t}$ algorithm is known to solve Max PC exactly on trees. (t = treewidth)
- We show that:
  - Max PC is W-hard parameterized by W, even for trees with $\Delta = 3$.
  - Max PC is W-hard parameterized by Δ, even for trees with $W = 4$.
  - Max PC is W-hard parameterized by t, even for $\Delta = W = 4$.
- $\rightarrow$ No $n^{o(\sqrt{\Delta} W t)}$ algorithm exists (assuming ETH).
- Strategy: Ind Set $\leq$ DNP $\leq$ Cap Max PC $\leq$ Max PC

Disjoint Neighborhood Packing

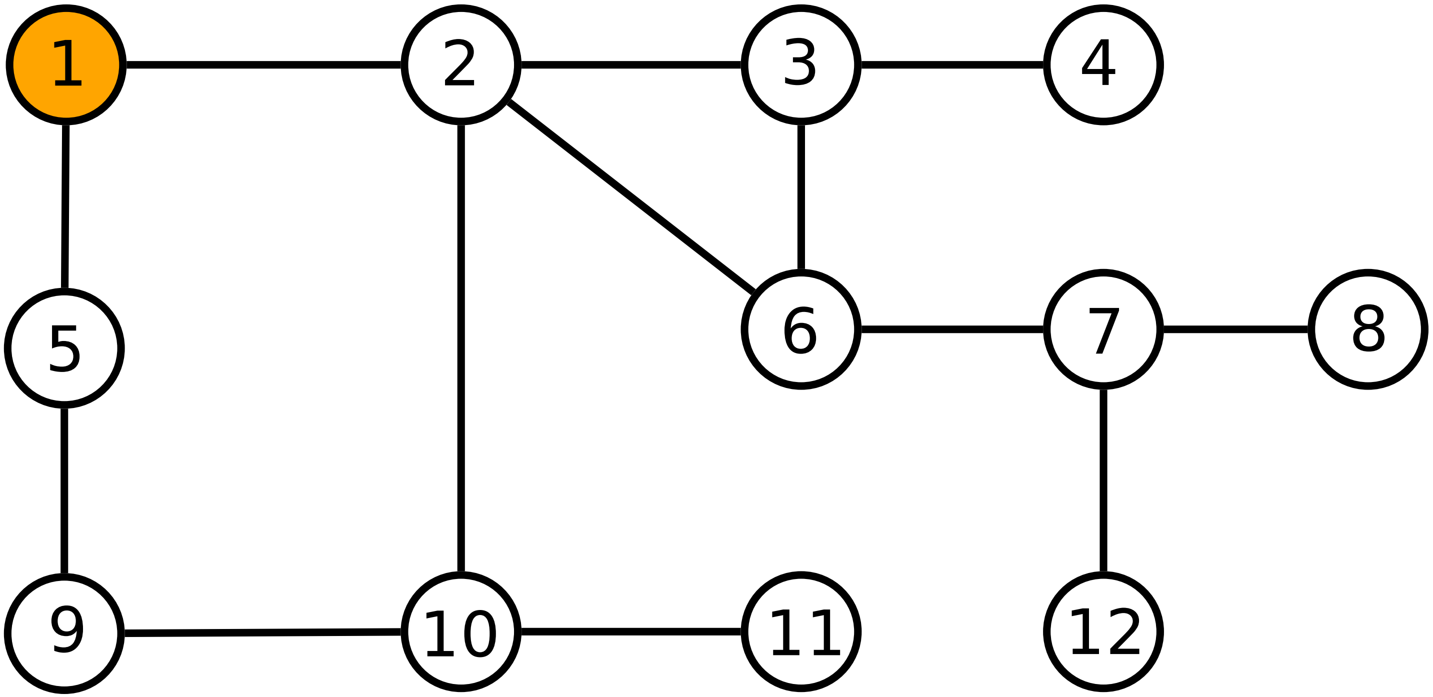
❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems



In this problem we want to select a maximum-size set of vertices such that all pairs have distance > 2

Disjoint Neighborhood Packing

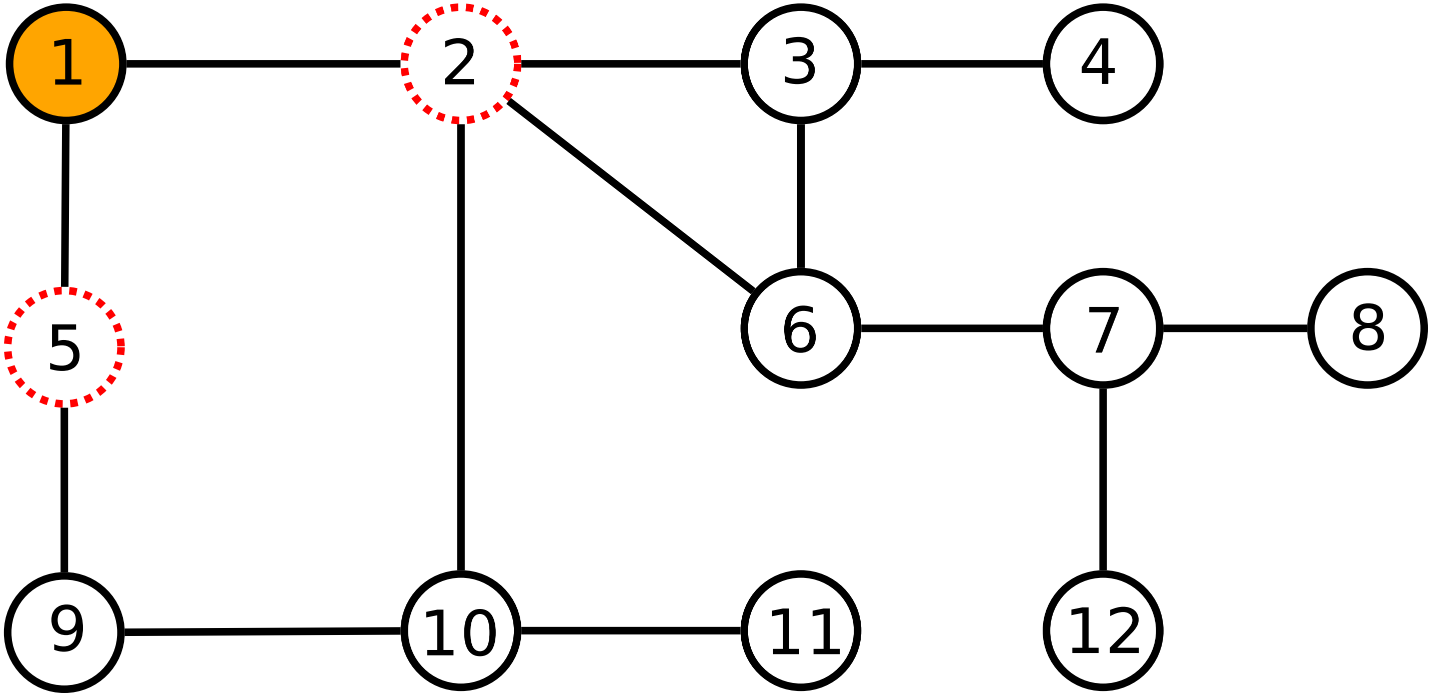
❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems



Selecting a vertex will disqualify . . .

Disjoint Neighborhood Packing

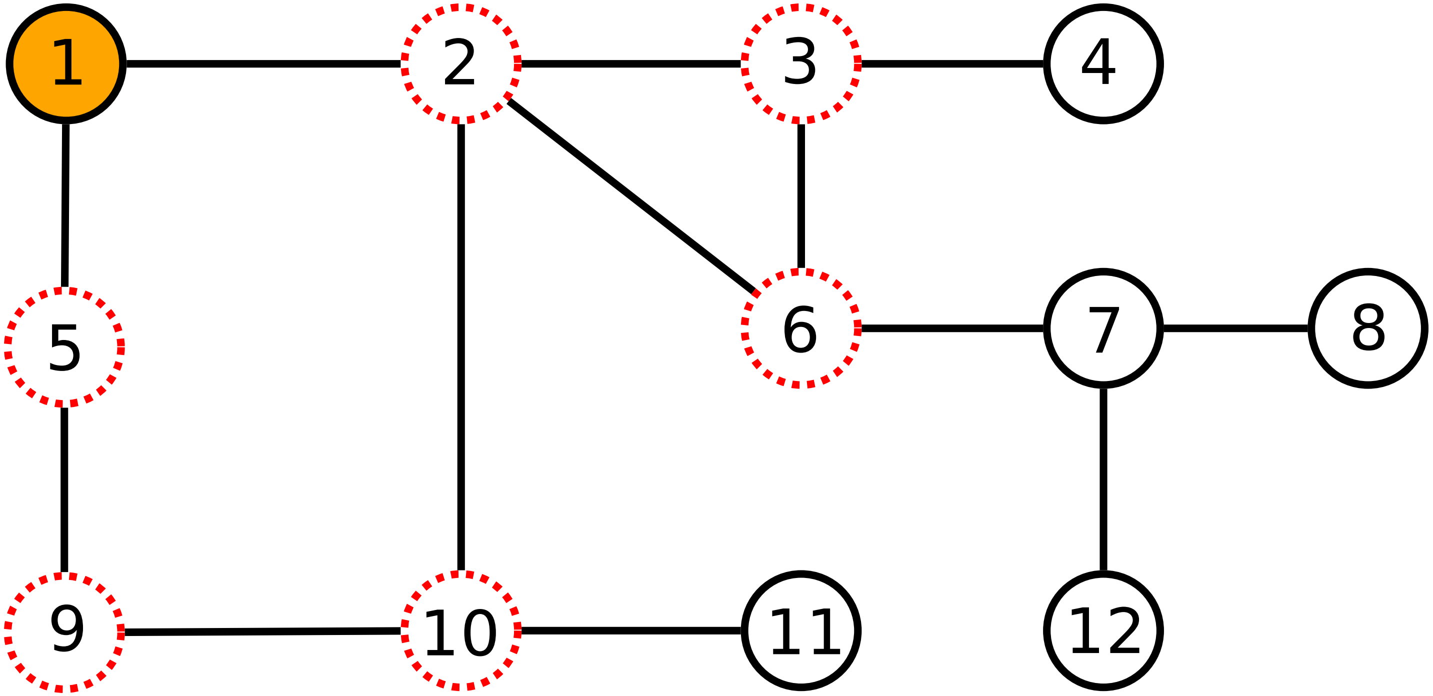
- ❖ Path Coloring
- ❖ Example
- ❖ Known results
- ❖ Edge slicing
- ❖ Max PC
- ❖ Max PC hardness results
- ❖ DNP
- ❖ Reduction
- ❖ Reduction
- ❖ Complexity jump
- ❖ $(p\Delta, pW, pT)$-MaxPC
- ❖ (pT)-MaxPC binary trees
- ❖ (pT)-MaxPC binary trees
- ❖ Algoritm cont'd
- ❖ Open problems



Selecting a vertex will disqualify its neighbors . . .

Disjoint Neighborhood Packing

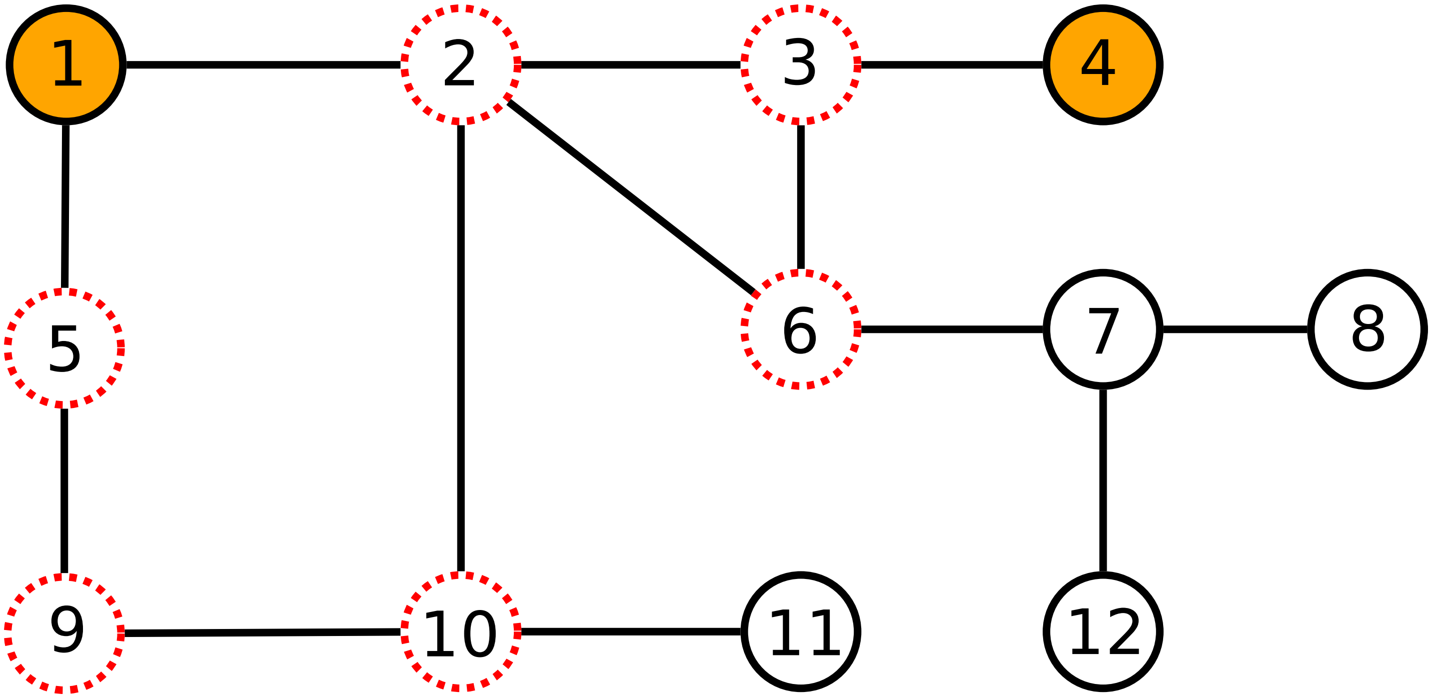
❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems



Selecting a vertex will disqualify its neighbors *and* their neighbors from future selection

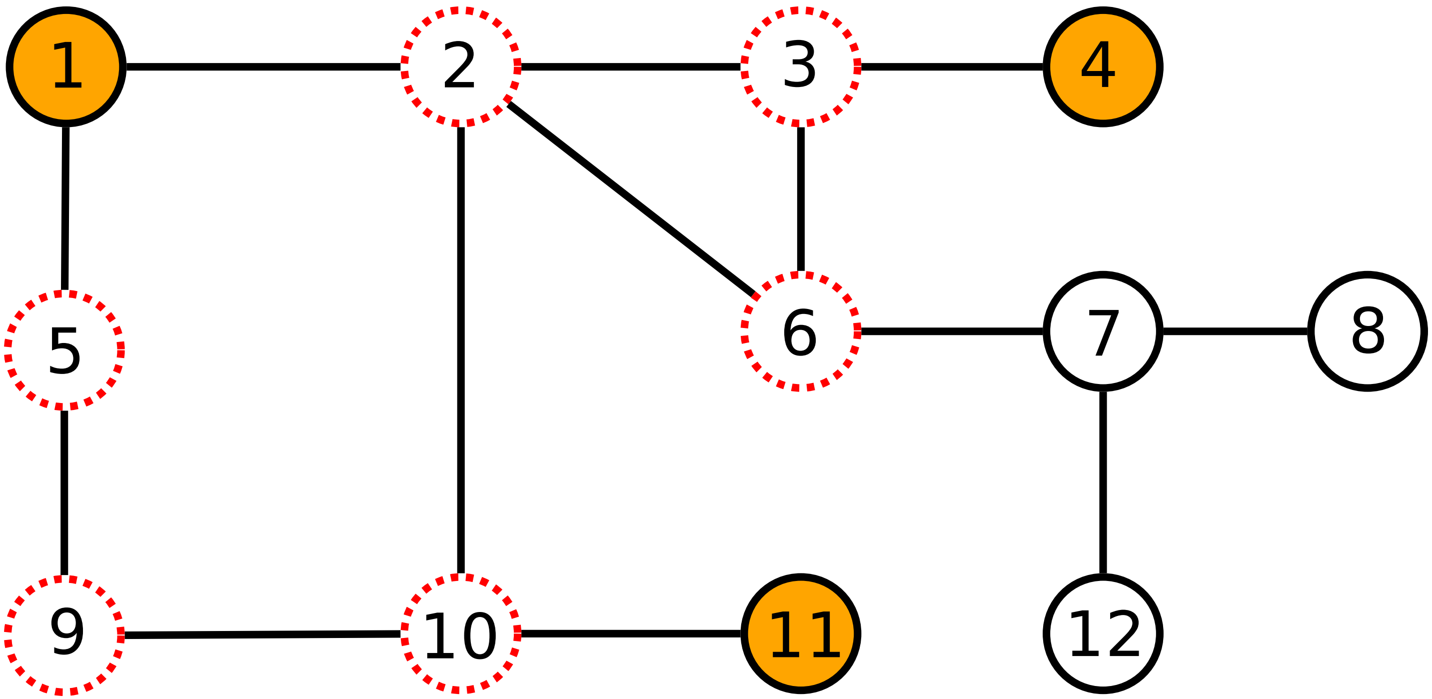
Disjoint Neighborhood Packing

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems



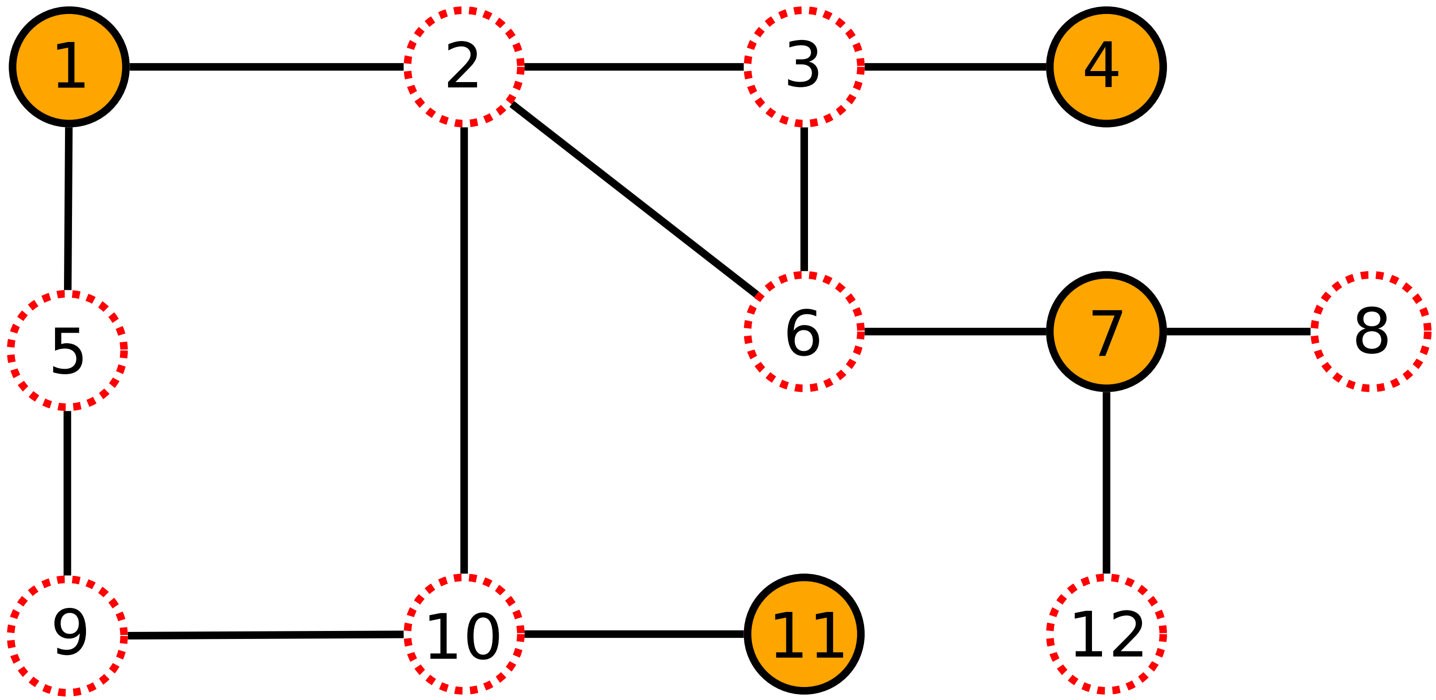
Disjoint Neighborhood Packing

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems



Disjoint Neighborhood Packing

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems

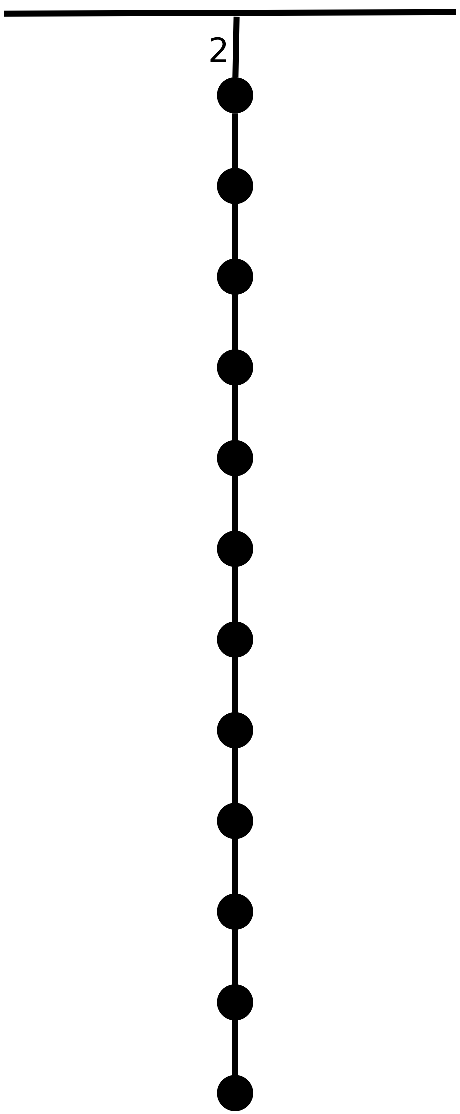


Problem is similar to Independent Set (and similarly W-hard)

Reduction

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

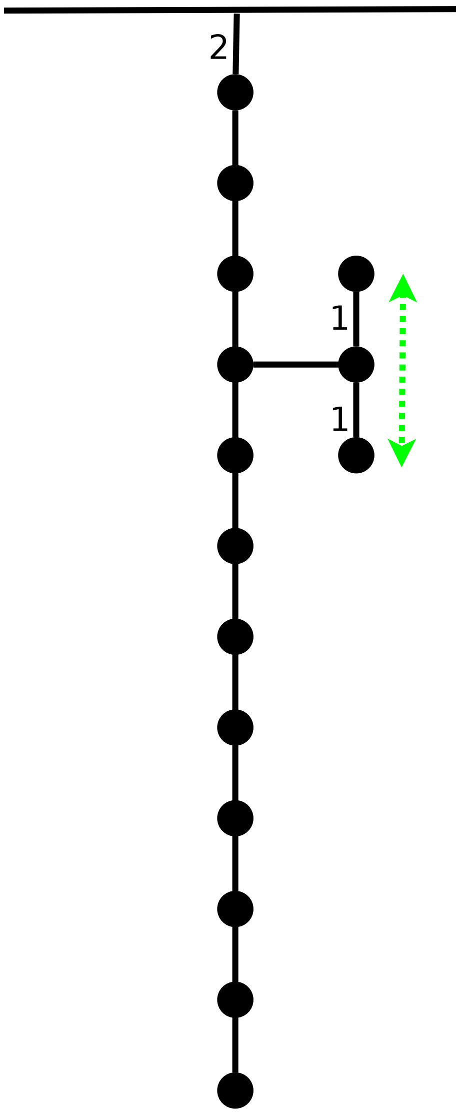
- Our basic gadget is a path on n vertices, to be attached to a “backbone” through an edge of capacity 2.



Reduction

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

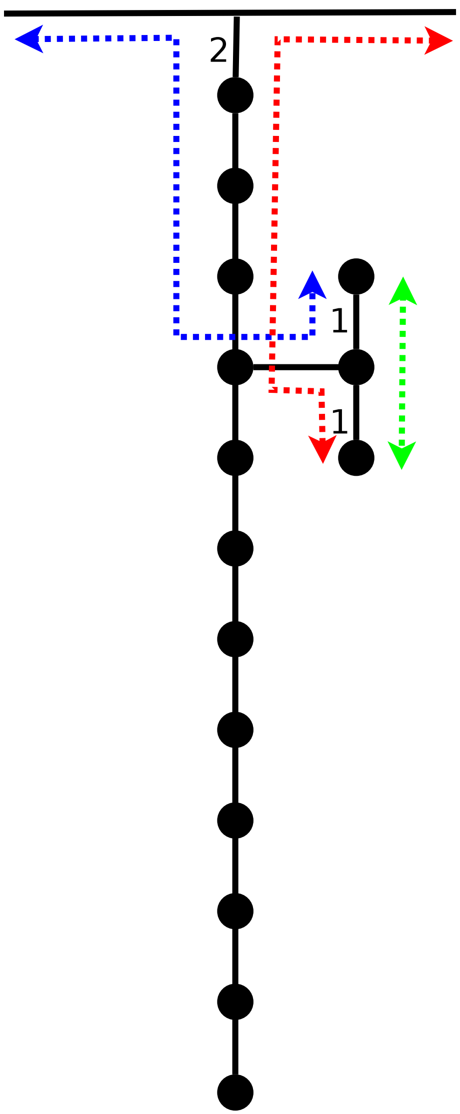
- To each vertex of a gadget we attach a short path with a local demand



Reduction

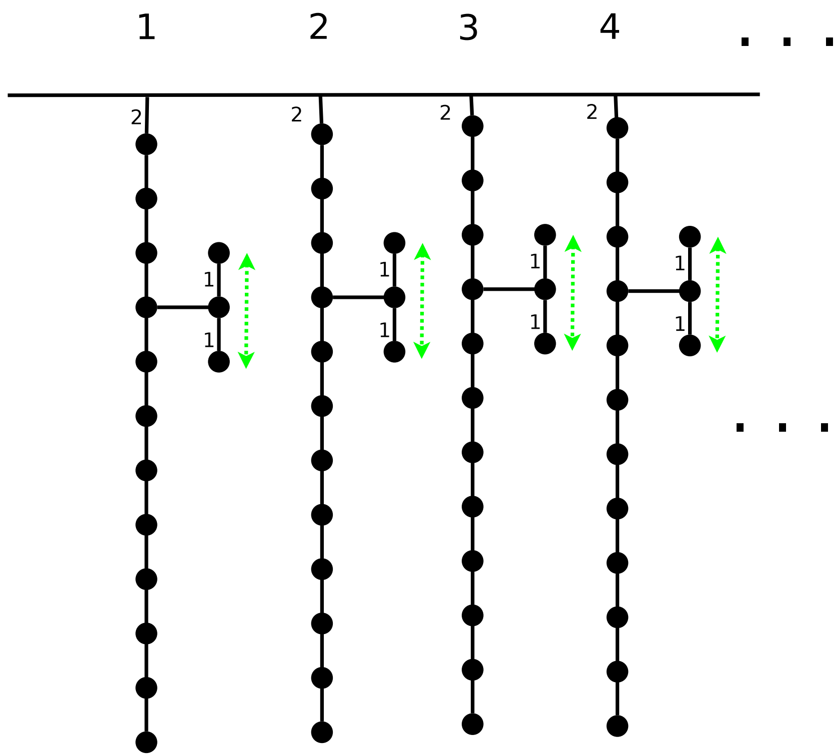
- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

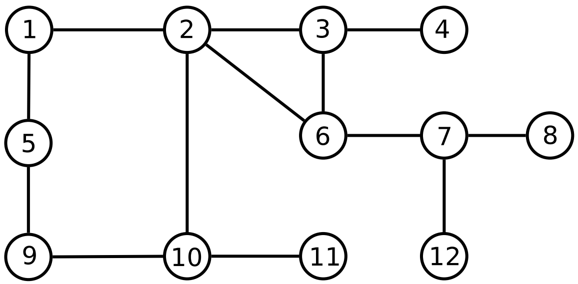
- The local demand will overlap with two global demands so that either the local demand is selected or both global demands



Reduction

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

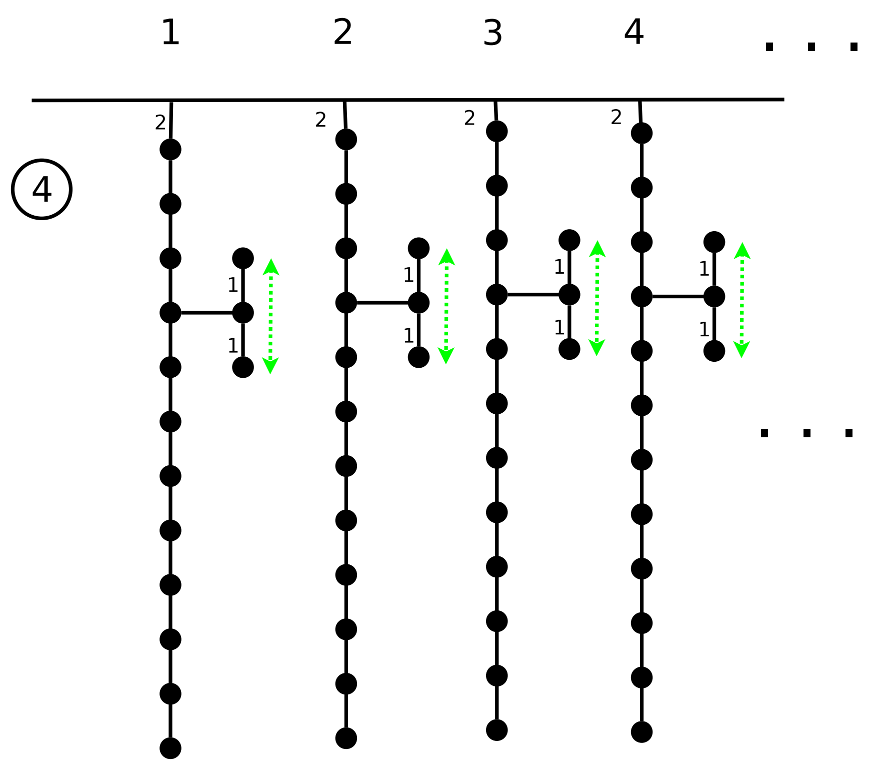


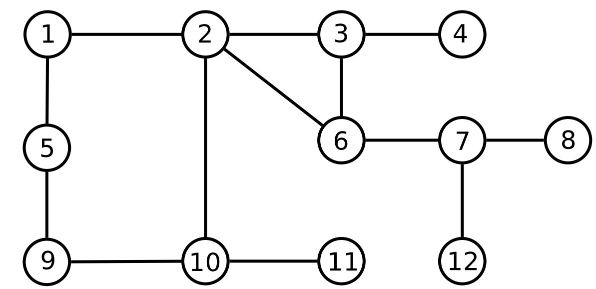


- We put n copies of the gadget on the backbone, one for each vertex.

Reduction

- ❖ Path Coloring
- ❖ Example
- ❖ Known results
- ❖ Edge slicing
- ❖ Max PC
- ❖ Max PC hardness results
- ❖ DNP
- ❖ Reduction
- ❖ Reduction
- ❖ Complexity jump
- ❖ $(p\Delta, pW, pT)$-MaxPC
- ❖ (pT)-MaxPC binary trees
- ❖ (pT)-MaxPC binary trees
- ❖ Algoritm cont'd
- ❖ Open problems

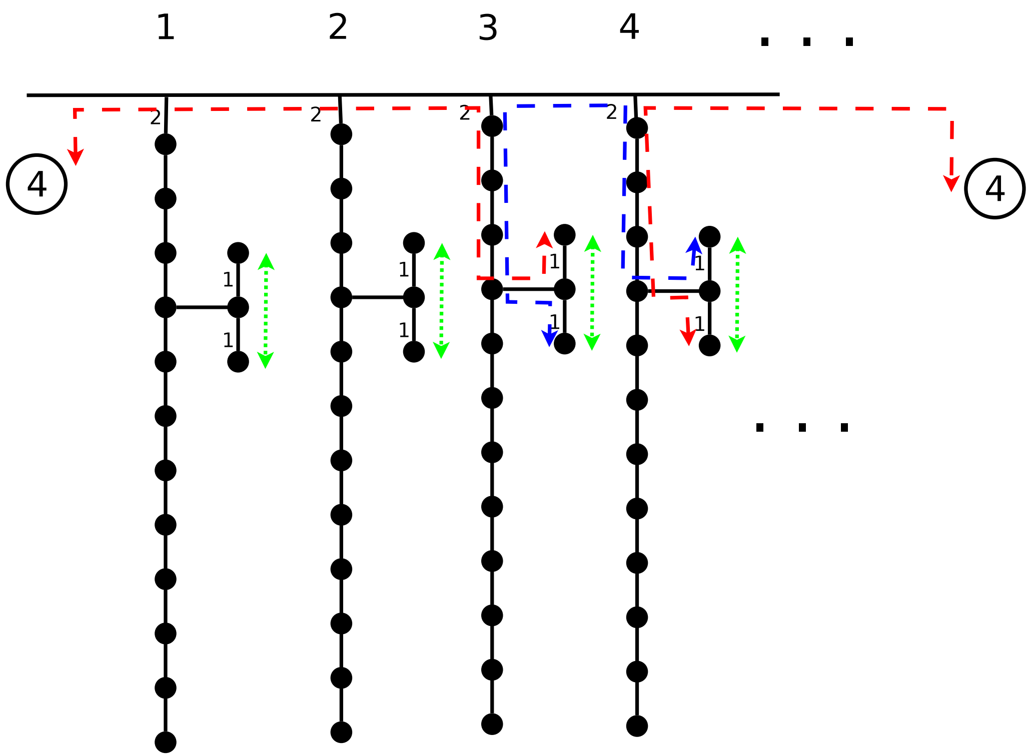


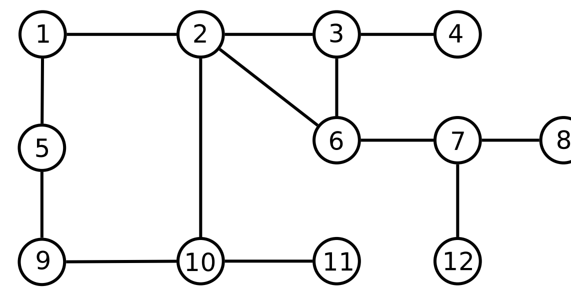


- For each vertex of the original graph we will make a set of global demands.

Reduction

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- **Reduction**
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

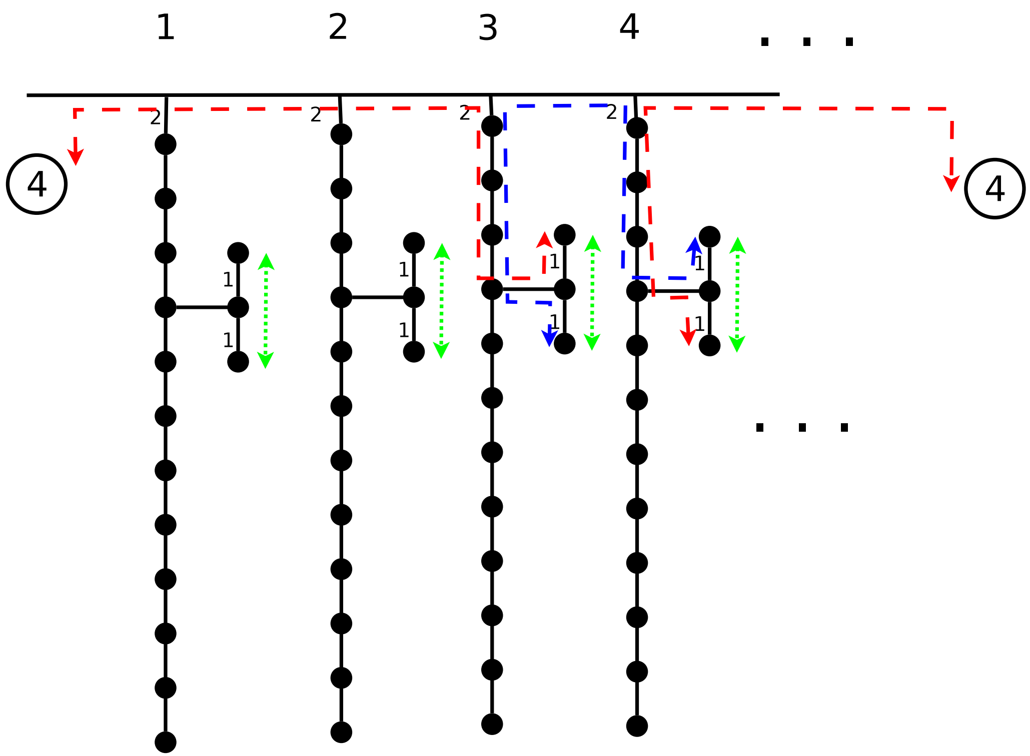


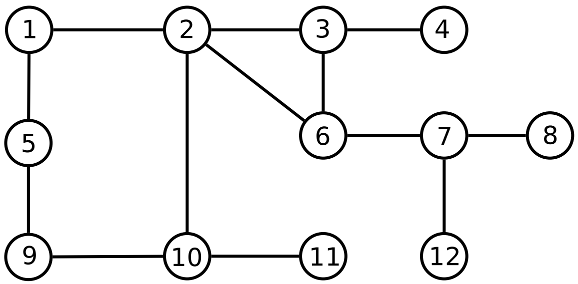


- The global demands use up all the neighbors' branches and are enough to increase the solution by one.

Reduction

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

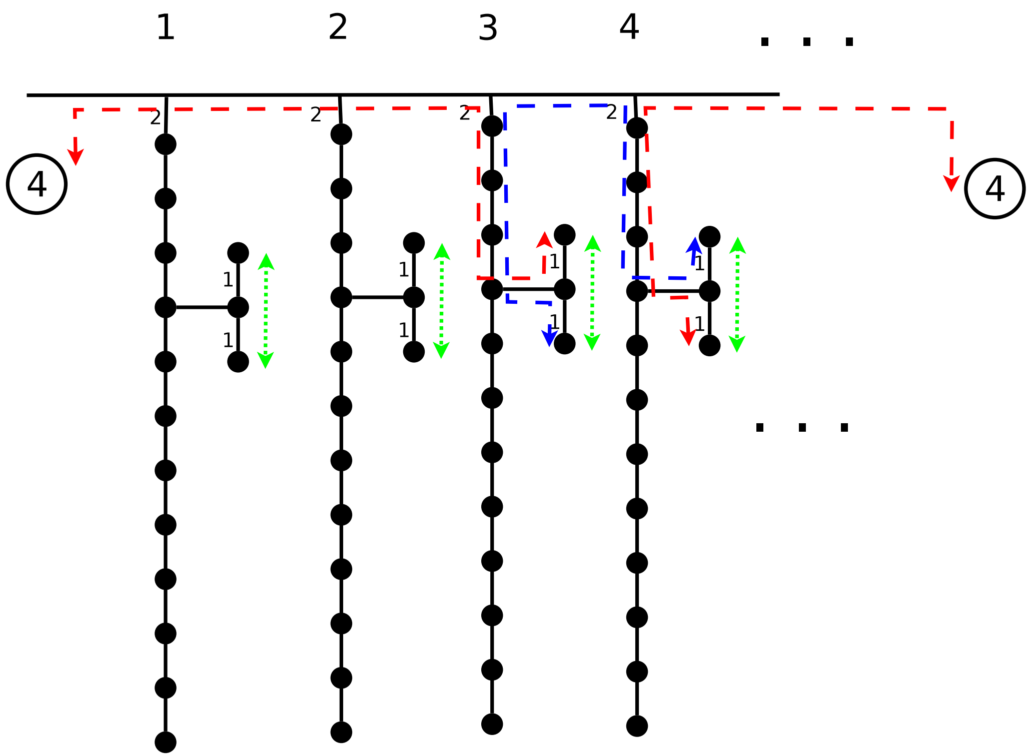


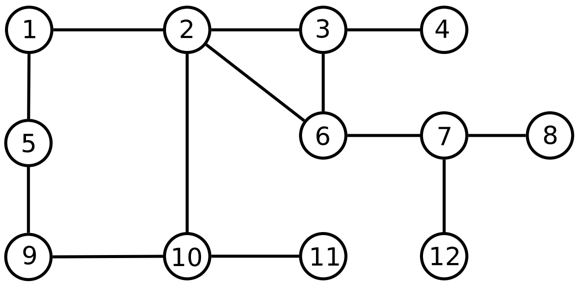


- If we can go from n^2 to $n^2 + k$ satisfied demands then original graph has DNP of size k.
- $\Delta = 3$ and $W = 2k$.

Reduction

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- **Reduction**
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

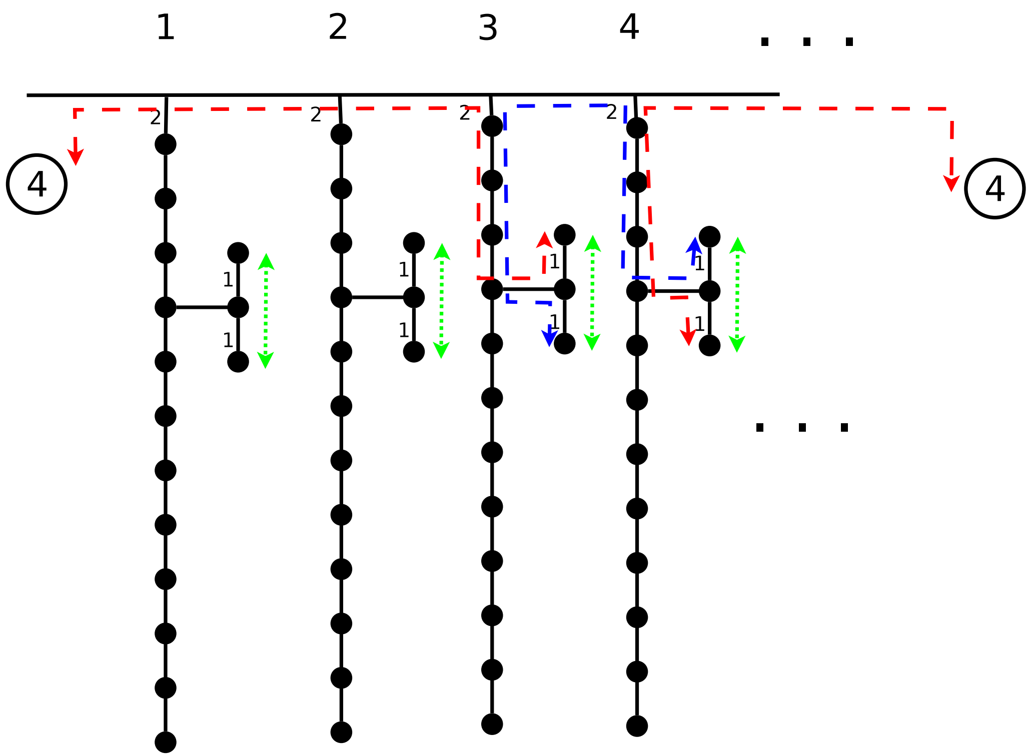


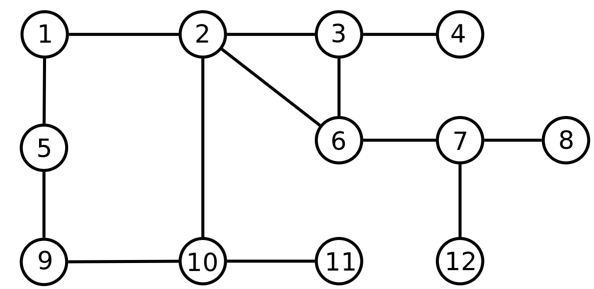


- Reduction for treewidth: Replace backbone with a $k \times n$ grid.

Reduction

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems





- Reduction for Δ: Replace backbone with a vertex of degree k^2. Use $\binom{k}{2}$ copies of this gadget to check compatibility between all pairs.

Complexity jump

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

- What makes Max PC harder than PC?
  - Intuitively, we first have to decide which requests to drop, then color the rest. The first part appears to be harder.
- What if we only want to drop a small number of requests T?

Complexity jump

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

- What makes Max PC harder than PC?
  - Intuitively, we first have to decide which requests to drop, then color the rest. The first part appears to be harder.
- What if we only want to drop a small number of requests T?

Another parameter is born. . .

$(p\Delta, pW, pT)$-*MaxPC*

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

- Recall that $(p\Delta, pW)$-PC is FPT
  - Bottom-up dynamic programming algorithm
- So, the problem is (essentially) to pick the dropped requests
- Observation: if more than $2\Delta W + T + 1$ requests touch a vertex reject immediately
  - Even if we drop T requests one of its incident edges will have $> W$ requests going through it.
- Otherwise, do bottom-up dynamic programming again.

(pT)-MaxPC binary trees

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems

MaxPC on undirected binary trees

- Slice edges until we are left with stars, solve PC on each star
- Some stars are "good", other "bad" (if all are good accept)
- If a sub-tree contains only good stars cut it off the tree
  - We can color everything there even if we don't drop any requests
- All leaf-stars are now bad

(pT)-MaxPC binary trees

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

- All leaf-stars are now bad
- All of them must be touched by a dropped request
- No dropped request can touch more than two leaves
  - If more than $2T$ leaf-stars reject
- Now graph has $O(T)$ leaf-stars and internal-stars ($\Delta = 3$)
- Easy to pick one endpoint of a dropped request. If we have $O(T)$ choices for the other endpoint $\rightarrow$ recursive $T^{O(T)}$ algorithm.

Algoritm cont'd

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

- Now graph has $O(T)$ leaf-stars and internal-stars ($\Delta = 3$)
- Easy to pick one endpoint of a dropped request. If we have $O(T)$ choices for the other endpoint $\rightarrow$ recursive $T^{O(T)}$ algorithm.
- Possible candidates are the $O(T)$ special stars and a (possible large) number of degree two stars.
  - But we can be greedy with degree two stars!
  - Pick the one that is furthest away

Algoritm cont'd

- Path Coloring
- Example
- Known results
- Edge slicing
- Max PC
- Max PC hardness results
- DNP
- Reduction
- Reduction
- Complexity jump
- $(p\Delta, pW, pT)$-MaxPC
- (pT)-MaxPC binary trees
- (pT)-MaxPC binary trees
- Algoritm cont'd
- Open problems

- Now graph has $O(T)$ leaf-stars and internal-stars ($\Delta = 3$)
- Easy to pick one endpoint of a dropped request. If we have $O(T)$ choices for the other endpoint $\rightarrow$ recursive $T^{O(T)}$ algorithm.
- Possible candidates are the $O(T)$ special stars and a (possible large) number of degree two stars.
  - But we can be greedy with degree two stars!
  - Pick the one that is furthest away
- The greedy part is the only part that requires $\Delta = 3$

Open problems

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems

Conclusions:

- PC is easy parameterized by Δ, W, but MaxPC is hard!
- In our reductions we have to drop many requests. If we parameterize also by T things get better.

What next?

- $(p\Delta, pT)$-MaxPC on undirected trees
- Other parameters?
- (pW)-MaxPC on rings?

The End

❖ Path Coloring
❖ Example
❖ Known results
❖ Edge slicing
❖ Max PC
❖ Max PC hardness results
❖ DNP
❖ Reduction
❖ Reduction
❖ Complexity jump
❖ $(p\Delta, pW, pT)$-MaxPC
❖ (pT)-MaxPC binary trees
❖ (pT)-MaxPC binary trees
❖ Algoritm cont'd
❖ Open problems

Thank you!

Questions?