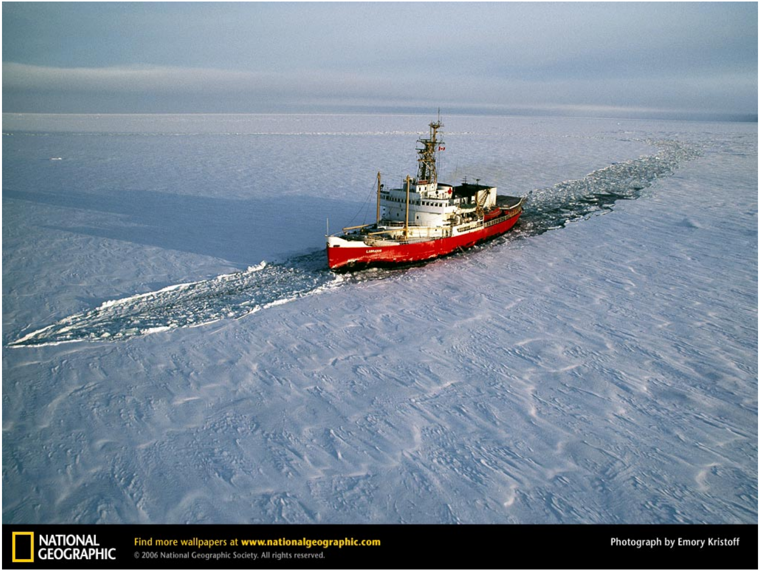
On the Complexity of Minimum Path Cover with Subpath Constraints for Multi-Assembly

Romeo Rizzi[1,*], Alexandru I. Tomescu[2,*], Veli Mäkinen[2]

[1] *Department of Computer Science, University of Verona, Italy*
[2] *Helsinki Institute for Information Technology HIIT, Department of Computer Science, University of Helsinki, Finland*
* *Equal contribution*

RECOMB-Seq 2014
31 March 2014


NATIONAL GEOGRAPHIC
Find more wallpapers at www.nationalgeographic.com
© 2006 National Geographic Society. All rights reserved.
Photograph by Emory Kristoff

MULTI-ASSEMBLY

Assembly of fragments from different, but related, sequences

- transcriptomics (RNA-Seq)
- viral quasi-species
- metagenomics

Assumptions:

✔ existing reference (genome-guided multi-assembly)

✖ no existing annotation (annotation-free)

OVERLAP AND SPLICING GRAPHS

Overlap graphs:

- reads $\equiv$ nodes
- overlaps $\equiv$ arcs
- $+$ coverage information

Splicing graphs:

- exons $\equiv$ nodes
- reads overlapping two exons $\equiv$ arcs
- $+$ coverage information

Existing reference $\Longrightarrow$ graphs are acyclic (DAGs)

Minimum Path Cover (MPC)

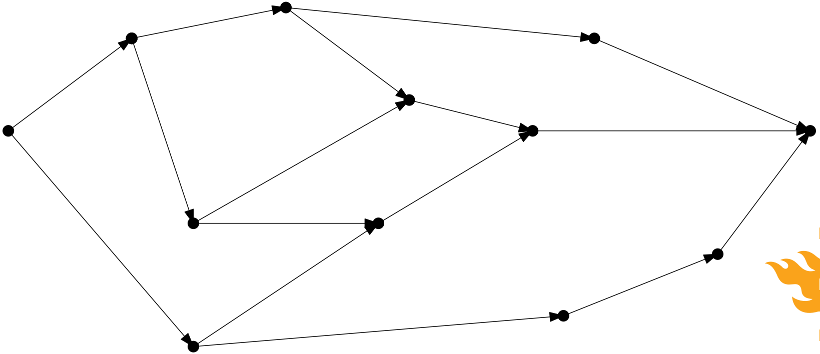
What is the minimum number of paths required to cover all nodes of a DAG?

- RNA-Seq: Cufflinks, CLASS, BRANCH
- Viral quasi-species: ShoRAH

Minimum Path Cover (MPC)

What is the minimum number of paths required to cover all nodes of a DAG?

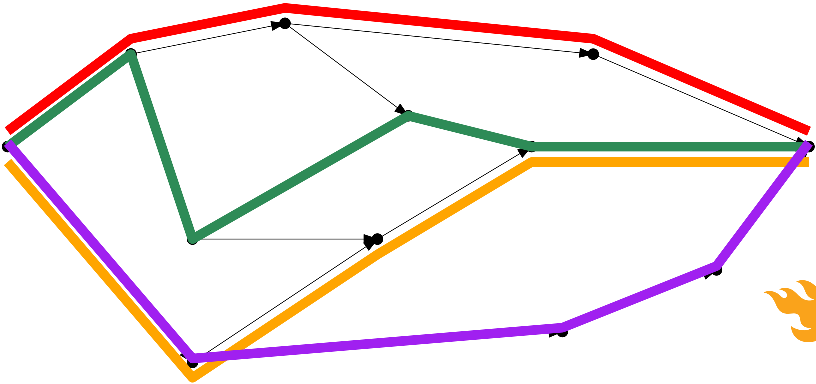
- RNA-Seq: Cufflinks, CLASS, BRANCH
- Viral quasi-species: ShoRAH



Minimum Path Cover (MPC)

What is the minimum number of paths required to cover all nodes of a DAG?

- RNA-Seq: Cufflinks, CLASS, BRANCH
- Viral quasi-species: ShoRAH



Minimum Path Cover (MPC)

In general it is NP-complete (one path iff G has a Hamiltonian path)

But it is solvable in polynomial-time on DAGs:

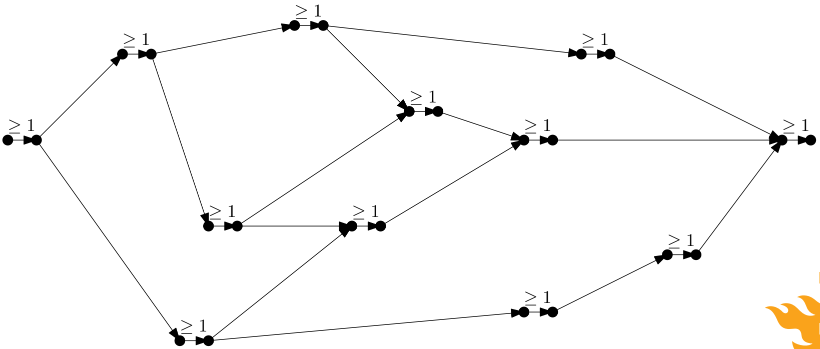
- Dilworth's theorem 1950 + Fulkerson's constructive proof 1956
- by a maximum matching algorithm, solvable in time $O(t(G)\sqrt{n})$
- the weighted version can be solved in time $O(n^2 \log n + t(G)n)$

where $t(G)$ is the number of arcs in the transitive closure of G.

MIN-COST MPC VIA MIN-COST FLOWS

- Unweighted case: MPC via Min-Flows, [*Pijls, Potharst, 2013*]
- Weighted case: MPC via Min-cost Flows

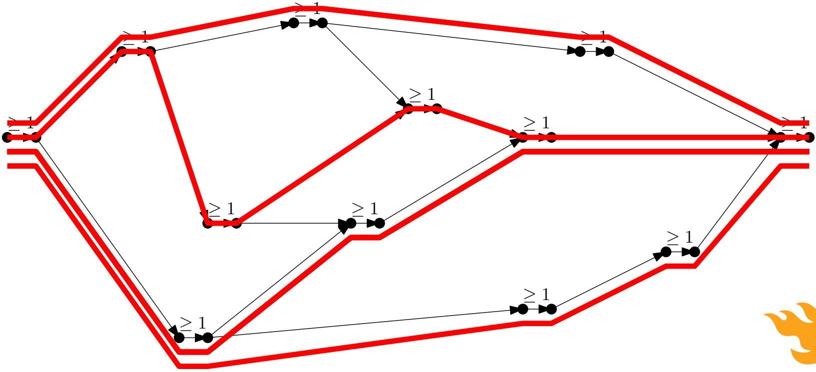
Assuming we know the minimum size of a path cover:



MIN-COST MPC VIA MIN-COST FLOWS

- Unweighted case: MPC via Min-Flows, [*Pijls*, *Potharst*, *2013*]
- Weighted case: MPC via Min-cost Flows

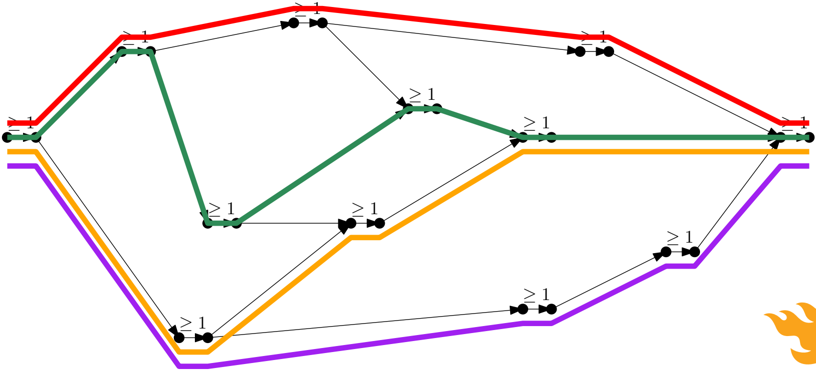
Assuming we know the minimum size of a path cover:



MIN-COST MPC VIA MIN-COST FLOWS

- Unweighted case: MPC via Min-Flows, [*Pijls, Potharst, 2013*]
- Weighted case: MPC via Min-cost Flows

Assuming we know the minimum size of a path cover:



MPC via Min-cost flows

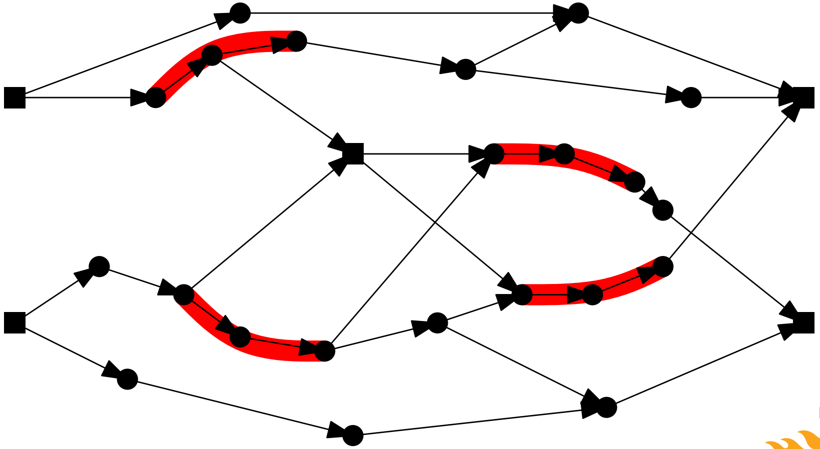
This flow problem can be reduced to a Min-cost circulation problem

- we add an arc from t to s with 'large' cost
- we have only demands ($= 1$)
- can be solved in time $O(n^2 \log n + nm)$ by [*Gabow and Tarjan, 1991*]

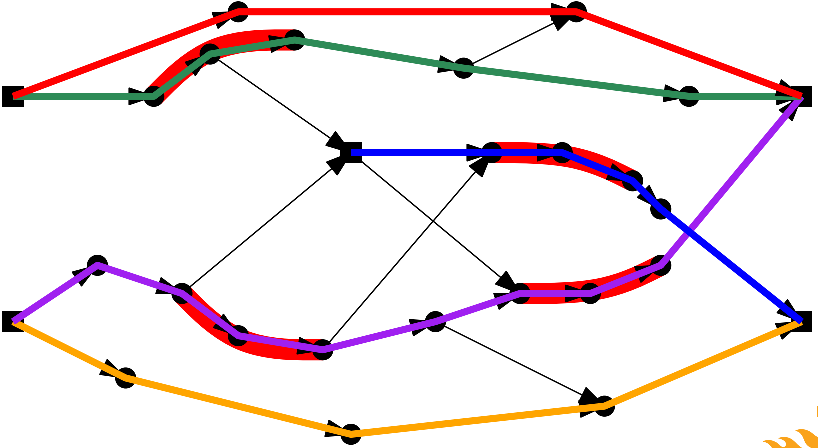
This is always better than $O(n^2 \log n + nt(G))$, because $m \leq t(G) \leq n^2$

- as soon as there is a path of length $O(n)$, we have $t(G) = O(n^2)$

Min-cost MPC with Subpath Constraints



Min-cost MPC with Subpath Constraints



MIN-COST MPC WITH SUBPATH CONSTRAINTS

INPUT: A DAG G and

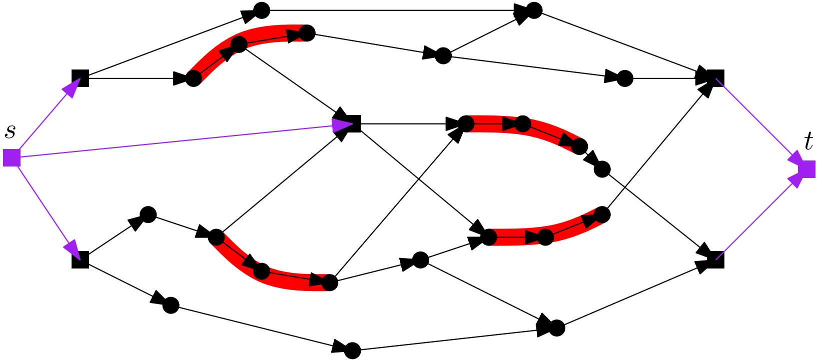
1. A superset S of the sources of G, and a superset T of the sinks of G
2. A cost $w(e)$ for each $e \in E(G)$
3. A family $\mathcal{P}^{in} = \{P_1^{in}, \ldots, P_t^{in}\}$ of directed paths in G

TASK: Find a minimum number k of directed paths $P_1^{sol}, \ldots, P_k^{sol}$ in G such that

1. Every node in $V(G)$ occurs in some P_i^{sol}
2. Every path $P^{in} \in \mathcal{P}^{in}$ is entirely contained in some P_i^{sol}
3. Every path P_i^{sol} starts in a node of S and ends in a node of T
4. $\sum_{i=1}^{k} \sum_{\text{edge } e \in P_i^{sol}} w(e)$ is minimum among all tuples of k paths satisfying 1.-3.

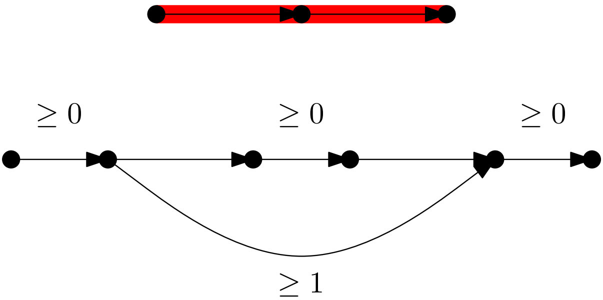
- introduced by [*Bao, Jiang, Girke, 2013, BRANCH*], but the case of overlapping constraints not solved

MIN-COST MPC WITH SUBPATH CONSTRAINTS



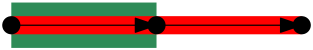
Min-cost MPC with Subpath Constraints

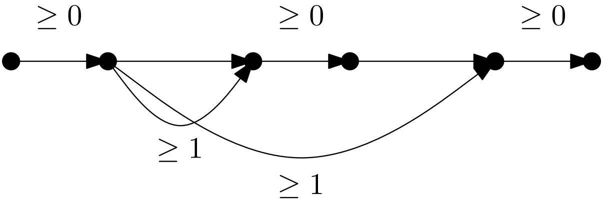
Subpath constraints as arc demands:



Min-cost MPC with Subpath Constraints

Problem 1: a constraint P included in another constraint Q

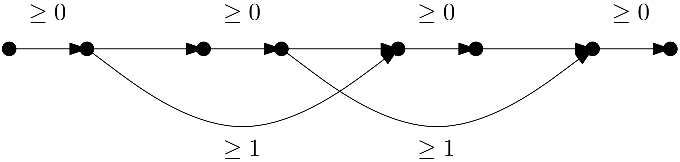




- Remove P
- Can be implemented in time $O(N)$ with a suffix tree for large alphabets, [*Farach, 1997*]
  - N = sum of lengths of Subpath Constraints

Min-cost MPC with Subpath Constraints

Problem 2: Suffix-prefix overlaps



- Iteratively merge constraints with longest suffix-prefix overlap
- All suffix-prefix overlaps can be found in optimal time $O(N + overlaps)$ by [*Gusfield, Landau and Schieber, 1992*]
- Our iterative merging also takes $O(N + overlaps)$ time

MIN-COST MPC WITH SUBPATH CONSTRAINTS

Pre-processing phase

- $O(N + c^2)$
  - *overlaps* $\leq c^2$

The flow problem can be reduced to a Min-cost circulation problem

- we add an arc from t to s with 'large' cost
- $O(n)$ nodes and $O(m + c)$ arcs
- only demands $(= 1)$

Min-cost MPC with Subpath Constraints can be solved in time $O(N + c^2 + n^2 \log n + n(m + c))$ by [*Gabow and Tarjan, 1991*]

MPC WITH PAIRED SUBPATH CONSTRAINTS

INPUT: A DAG G and

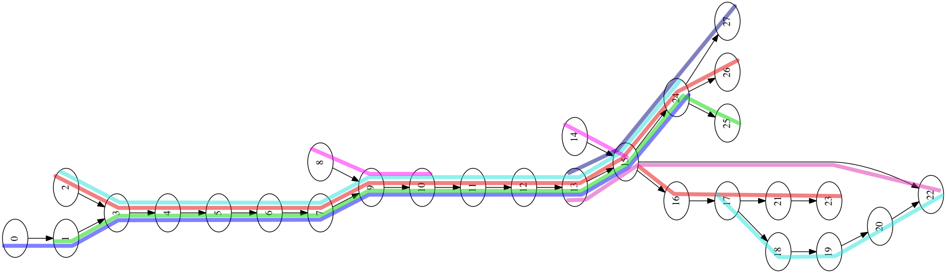
1. A family $\mathcal{P}^{in} = \{(P^{in}_{1,1}, P^{in}_{1,2}), \ldots, (P^{in}_{t,1}, P^{in}_{t,2})\}$ of pairs of directed paths in G

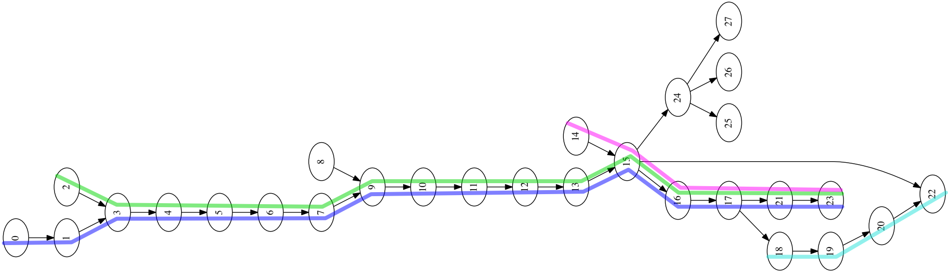
TASK: Find a minimum number k of directed paths $P^{sol}_1, \ldots, P^{sol}_k$ in G such that

1. Every node in $V(G)$ occurs in some P^{sol}_i
2. For every pair $(P^{in}_{j,1}, P^{in}_{j,2}) \in \mathcal{P}^{in}$, there exists P^{sol}_i such that both $P^{in}_{j,1}$ and $P^{in}_{j,2}$ are entirely contained in P^{sol}_i

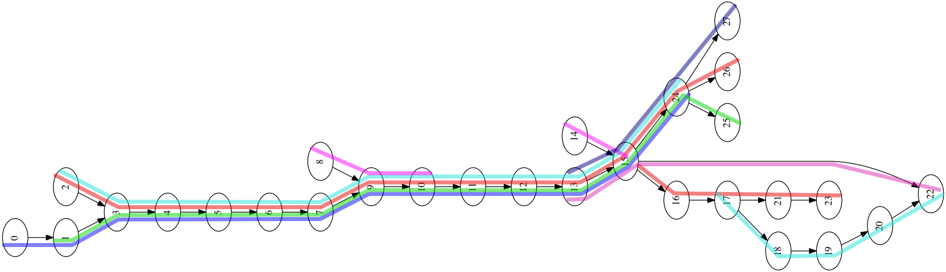
- introduced by [*Song and Florea, 2013, CLASS*]
- we show that it is
  - NP-hard; not FPT when parametrized by k
  - FPT in the number of constraints and nodes that need to be covered
- solved in parallel by [*Beerenwinkel, Beretta, Bonizzoni, Dondi and Pirola, 2014*]

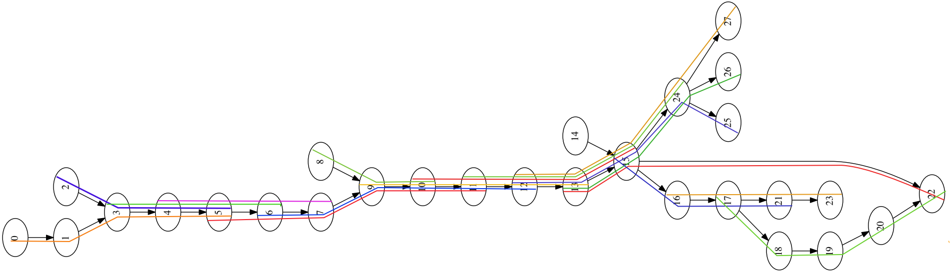
Gene AKT3 - Annotation and Cufflinks



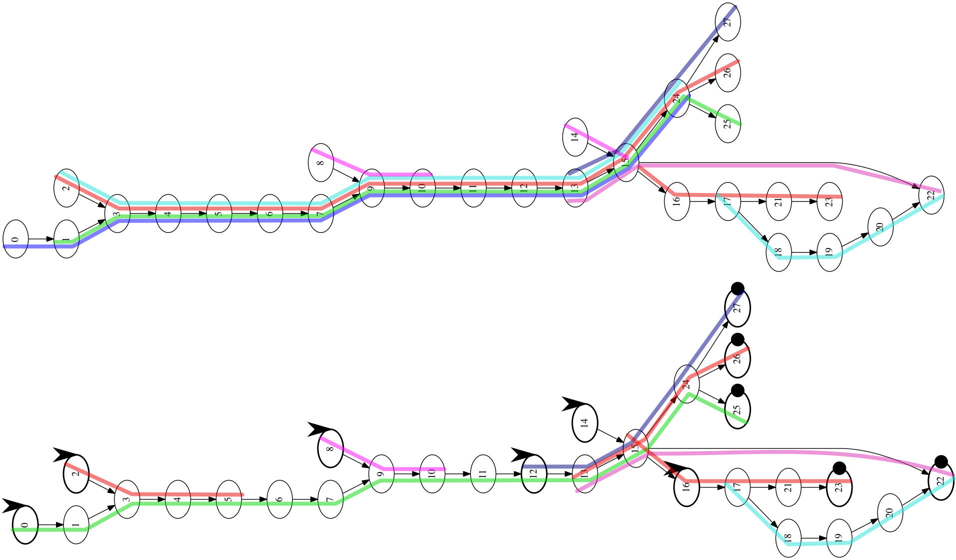


Gene AKT3 - Annotation and Subpaths

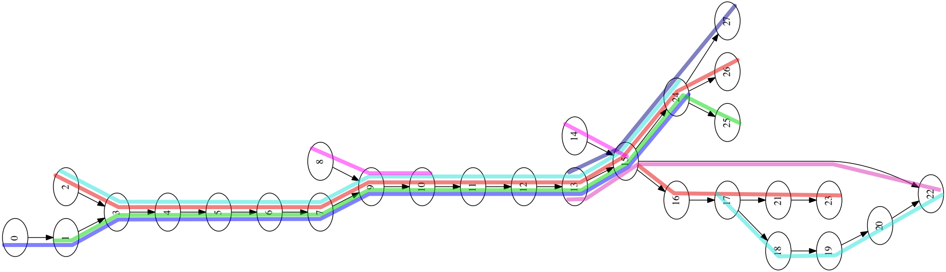


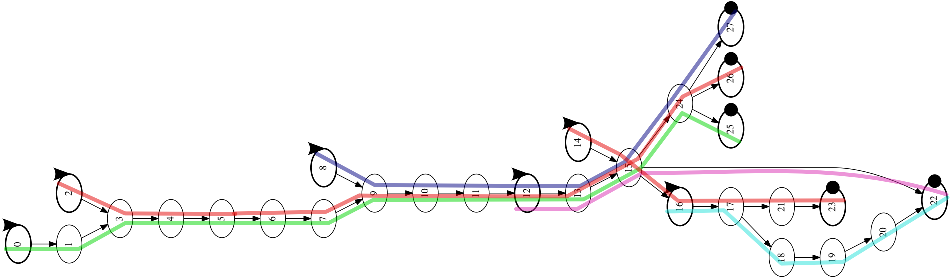


Gene AKT3 - Annotation and Merged subpaths



Gene AKT3 - Annotation and MPC-SC





Conclusions

- Min-cost Minimum Path Cover

$$O(n^2 \log n + nm)$$

- Min-cost Minimum Path Cover with Subpath Constraints

$$O(N + c^2 + n^2 \log n + n(m + c))$$

  - c = number of Subpath Constraints
  - N = sum of lengths of Subpath Constraints

- Minimum Path Cover with Pairs of Subpaths Constraints

NP-hard, but FPT in the total number of constraints

- Future work: a better integration of observed coverages
- Implementation for RNA-Seq reads under way

ACKNOWLEDGEMENTS

Partial support by

- Academy of Finland — Centre of Excellence in Cancer Genomics Research (grant 250345)
- Finnish Cultural Foundation

Romeo Rizzi

Veli Mäkinen

Thanks to

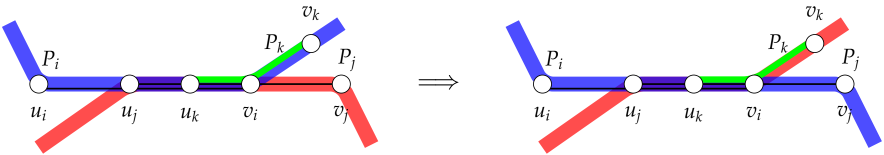
- Anna Kuosmanen and Ahmed Sobih for preliminary implementation and experiments

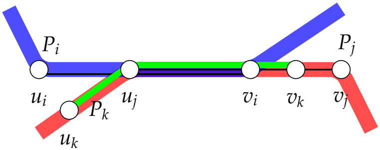
Thank you!

Pictorial proof of Step 2.

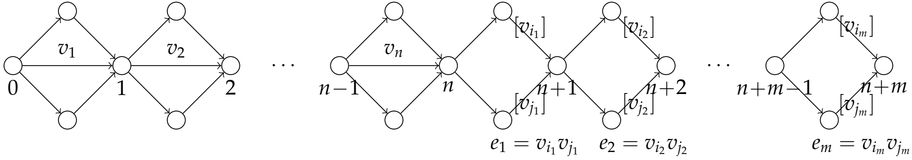
Lemma

Step 2. does not increase the cardinality of the solution path cover.





NP-completeness of Problem MPC-PSC



Theorem

Problem MPC-PSC is NP-complete.

- A graph $G = (\{v_1, \dots, v_n\}, \{e_1, \dots, e_m\})$ has chromatic number 3 iff the DAG above admits a solution with 3 paths.

Corollary

For no $\varepsilon > 0$ there exists a $\left(\frac{4}{3} - \varepsilon\right)$-approximation algorithm for Problem MPC-PSC unless P=NP. Moreover, the problem is not FPT when parameterized on OPT (the minimum number of paths in a solution).

Problem MPC-PSC is FPT in the total number of constraints

Lemma

Let C be a set of constraints on a DAG. There exists a directed path P in G which satisfies all constraints in C iff any two constraints in C are compatible.

Theorem

Given an instance for Problem MPC-PSC, we can decide in polynomial time if $OPT = 2$, and if so, find the two solution paths. Moreover, Problem MPC-PSC is fixed-parameter tractable (FPT) in the total number C of input constraints.

- construct the 'in-compatibility' graph; this is bipartite iff $OPT = 2$
- partition the set of constraints in all possible ways and check that all constraints in every class are pairwise compatible