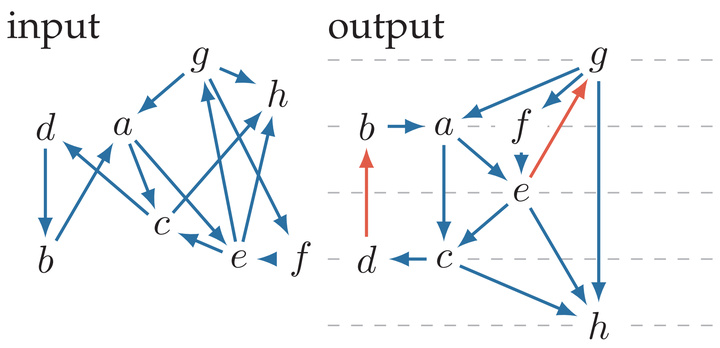
FASTER WAY TO AGONY

NIKOLAJ.TATTI@AALTO.FI
Aalto University, Helsinki Institute of Information Technology

DISCOVERING HIERARCHY



Given a directed graph $G = (V, E)$, assign to each vertex an integer rank minimizing

$$\sum_{(u,v)\in E} q(r(u), r(v)) \quad .$$

CHOOSING A SCORE

The score should penalize backward edges. If we set

$$q(x, y) = I[x \geq y],$$

then optimal solution is equal to FAS (APX-hard).

Set

$$q(x, y) = \max(y - x + 1, 0),$$

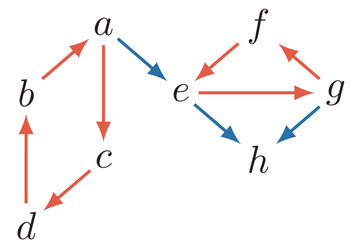
- vertices with the same rank, penalize by 1
- (u, v) with $r(u) = r(v) + 1$, penalize by 2
- ...

Optimization can be done in $O(nm^2)$ time (Gupte et al., 2011)
We improve the running time to $O(m^2)$.

EULERIAN SUBGRAPHS

Assume a graph $G = (V, E)$

Eulerian subgraph (V, F)
$=$ out-degree is equal to in-degree.



THEOREM Let r be *any* ranking. Let (V, F) be *any* eulerian subgraph. Then $q(r, G) \geq |F|$.

COROLLARY If $q(r, G) = |F|$,
then r is optimal.

DUALITY GAP

maximal eulerian subgraph
$=$ residual graph is DAG

rank conforming to the eulerian subgraph
$=$ all edges in residual graphs are forward

Define slack of an edge

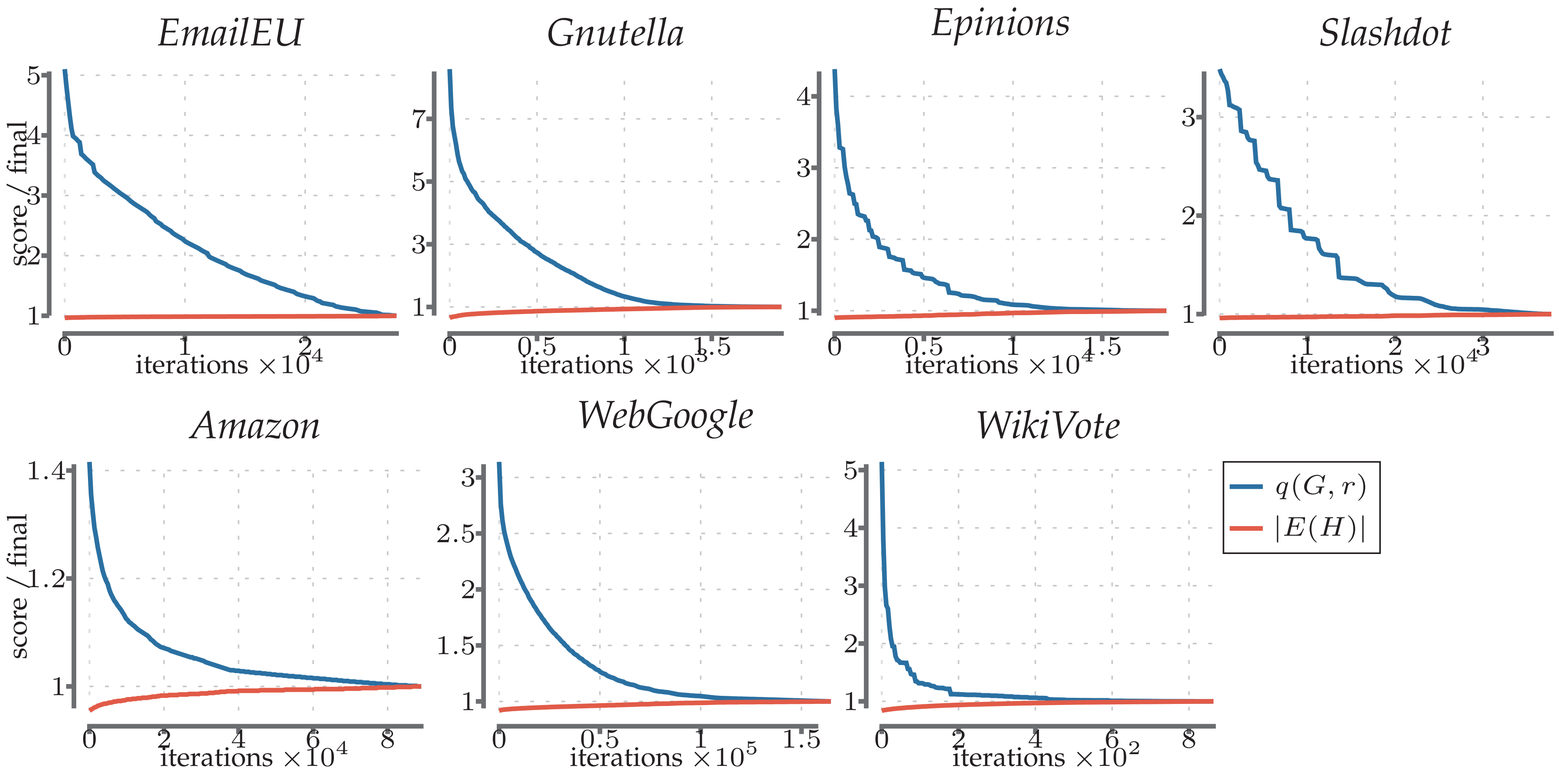
$$sl((u, v), r) = \max(r(v) - r(u) - 1, 0) \quad .$$

THEOREM let (V, F) be maximal eulerian subgraph, r conforming rank. Then

$$q(r, G) - |F| = \sum_{e\in F} sl(e, r) \quad .$$

EXPERIMENTS

Dataset	$\lvert V\rvert$	$\lvert E\rvert$	iterations	gap	agony	time	baseline
Amazon	403 394	3 387 388	89 046	911 095	1 973 965	4h27m	–
Gnutella	62 586	147 892	1 907	150 851	18 964	45s	20m
EmailEU	265 214	418 956	27 679	500 177	120 874	2m	3h45m
Epinions	75 879	508 837	18 652	922 817	264 995	20m	1h40m
Slashdot	82 168	870 161	37 858	1 891 586	748 582	1h5m	7h3m
WebGoogle	875 713	5 105 039	164 708	4 110 696	1 841 215	2h32m	–
WikiVote	7 115	103 689	865	76 149	17 676	7s	1m

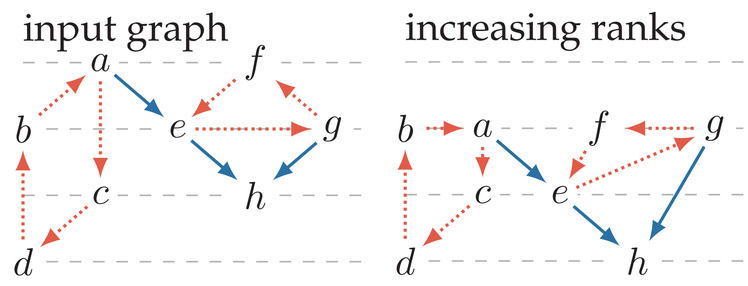


ALGORITHM

```
1  Find maximal eulerian subgraph H;
2  Find conforming rank r (O(m) time);
3  while there is slack do
4      (u, v) ← edge in H with slack;
5      r(u) ← r(v) − 1;
6      check that forward edges in
       residual DAG remain forward;
7      check that slack is not increased;
8      if r(v) is not changed then
9          do nothing (Case 1);
10     if r(v) is changed then
11         modify subgraph (Case 2);
```

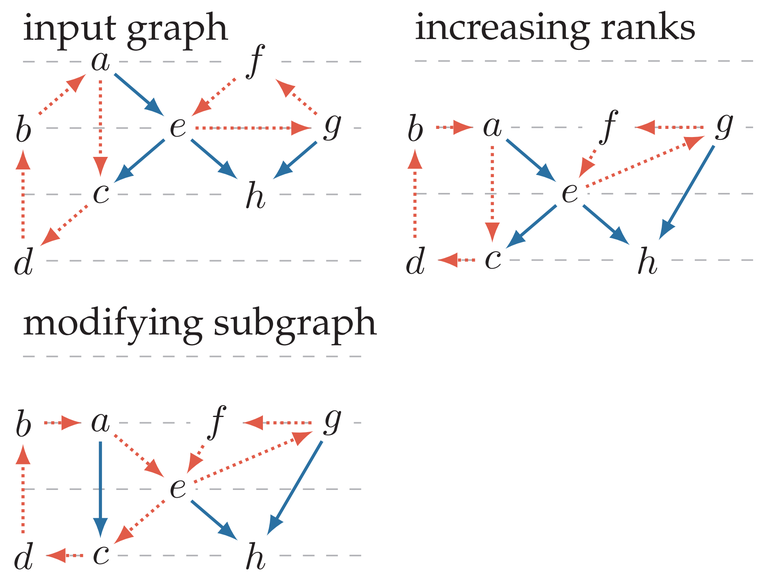
CASE 1

increasing $r(a)$ did not increased $r(c)$



CASE 2

increasing $r(a)$ increased $r(c)$



There is a path from u to v

- edges are in residual graph with 0 slack
- or in eulerian subgraph (with reversed direction)

Modify graph:

- add residual edges
- remove eulerian edges (also (u, v))

COMPLEXITY

A single step can be done in $O(m)$ time

Each step reduces the number of non-slack edges at most m steps.

Computational complexity, $O(m^2)$