On Digraph Width Measures in Parametrized Algorithmics

R. Ganian, P. Hliněný, J. Kneis, A. Langer, J. Obdržálek, P. Rossmanith

Masaryk University, Brno and RWTH Aachen

IWPEC'09 Copenhagen, September 10

The results

Problem	K-width	DAG-depth	DAG-width	Cycle-rank	DAG	Bi-rank-width
HAM	**FPT**	**FPT**	XP * W[2]-hard	XP *	P	XP W[2]-hard
c-PATH	**FPT**	**FPT**	XP *	XP *	P	**FPT**
k-PATH	**p-NPC**	**p-NPC**	**NPC**	**NPC**	**NPC**	*open*
DIDS	**p-NPC**	**p-NPC**	**NPC**	**NPC**	**NPC**	**FPT**
DISTP	**p-NPC**	**p-NPC**	**NPC**	**NPC**	**NPC**	**FPT**
MAXDICUT	p-NPC	p-NPC	NPC	NPC	NPC	**XP**
c-OCN	**p-NPC**	**p-NPC**	NPC	NPC	NPC	**FPT**
DFVS	*open*	*open*	p-NPC	p-NPC	P	**FPT**
KERNEL	p-NPC	p-NPC	p-NPC	p-NPC	P	**FPT**
φ-MSO$_1$MC	**p-NPH**	**p-NPH**	NPH	NPH	NPH	**FPT**
φ-LTLMC	**p-coNPH**	**p-coNPH**	**coNPH**	**coNPH**	**coNPC**	**p-coNPH**
PARITY	XP	XP	XP *	XP *	P	XP

Tree-width – a success story

Tree-width

- Robertson and Seymour (1984)
- connectivity measure for *undirected* graphs
- measures how close a graph is to being a tree
- cops-and-robber game characterization

The success story

- Many NP-hard become tractable on graphs of bounded tree-width: COLOURING, $MSOL_1$MC, HAMILTONIANPATH, K-CLIQUE . . .
- Some problems of "unknown" complexity become tractable as well: GRAPHISOMORPHISM, PARITYGAMES . . .
- Tree-width (and tree-decomposition) can be computed in FPT.

Directed graph measures I

The motivation

- DAGs are “easy”
- But what about oriented cliques?
- Some problems are naturally defined on directed graphs.

The tree-width like measures

- directed tree-width [JRST98/01]
- D-width [Saf05]
- DAG-width [O.06],[BDHK06]
- Kelly-width [HK07]
- . . .

Directed graph measures I

The motivation

- DAGs are “easy”
- But what about oriented cliques?
- Some problems are naturally defined on directed graphs.

The tree-width like measures

- directed tree-width [JRST98/01]
- D-width [Saf05]
- DAG-width [O.06],[BDHK06]
- Kelly-width [HK07]
- . . .

Directed graph measures II

Some features

- Cops-and-robber game characterization.
- Coincide with tree-width if we forget the edge orientation.
- Many of these measures are hard to compute.
- Low on DAGs.

Some other measures

- directed path-width [RST05, Bar06]
- cycle-rank [Egg63]
- clique-width [CO00]
- bi-rank-width [Kan08]

Directed graph measures II

Some features

- Cops-and-robber game characterization.
- Coincide with tree-width if we forget the edge orientation.
- Many of these measures are hard to compute.
- Low on DAGs.

Some other measures

- directed path-width [RST05, Bar06]
- cycle-rank [Egg63]
- clique-width [CO00]
- bi-rank-width [Kan08]

Problems defined on directed graphs

The good . . .

- HAMILTONIANPATH, K-PATH etc. are *polynomial* (XP) on bounded directed tree-width, DAG-width and Kelly-width.
- PARITY is *polynomial* on bounded DAG-width and Kelly-width

. . . the bad . . .

- HAMILTONIANPATH is *W[2]-hard* on DAG-width [LKM08]
- DFVS,KERNEL,MAXDICUT are *NP-hard* on graphs of low directed widths [KO08,LKM08]

. . . and the ugly

- Some problems are hard even on DAGs!
- The decompositions are hard to compute.

Problems defined on directed graphs

The good . . .

- HAMILTONIANPATH, K-PATH etc. are *polynomial* (XP) on bounded directed tree-width, DAG-width and Kelly-width.
- PARITY is *polynomial* on bounded DAG-width and Kelly-width

. . . the bad . . .

- HAMILTONIANPATH is *W[2]-hard* on DAG-width [LKM08]
- DFVS,KERNEL,MAXDICUT are *NP-hard* on graphs of low directed widths [KO08,LKM08]

. . . and the ugly

- Some problems are hard even on DAGs!
- The decompositions are hard to compute.

Problems defined on directed graphs

The good . . .

- HAMILTONIANPATH, K-PATH etc. are *polynomial* (XP) on bounded directed tree-width, DAG-width and Kelly-width.
- PARITY is *polynomial* on bounded DAG-width and Kelly-width

. . . the bad . . .

- HAMILTONIANPATH is *W[2]-hard* on DAG-width [LKM08]
- DFVS,KERNEL,MAXDICUT are *NP-hard* on graphs of low directed widths [KO08,LKM08]

. . . and the ugly

- Some problems are hard even on DAGs!
- The decompositions are hard to compute.

So what?

The ideal measure

- Would be small on many interesting classes of graphs.
- Many important problems should become FPT (or at least XP) when parametrized by this measure.
- Ideally would be FPT to compute.

To find it, we need to

- Find out what makes problems hard on DAGs, and graphs of low width.
- Compare the known measures on a wide range of relevant and interesting problems.

DAG-depth (ddp)

Inspired by tree-depth of Nešetřil and Ossona de Mendez.

$$ddp(G) = \begin{cases} 1 & \text{if } |V(G)| = 1 \\ 1 + \min_{v \in V(G)} ddp(G \setminus v) & \text{if there is a single maximal } G_v \\ \max_{v \in V(G)} ddp(G_v) & \text{otherwise} \end{cases}$$

Where G_v is the subgraph of G reachable from v.

Some features

- game characterization (lift-free strategy for cops)
- at least as big as DAG-width and cycle-rank
- can be arbitrarily large on DAGs
- $\lfloor \log_2 l \rfloor < ddp(G) \leq l$, where l is the length of the longest path in G

K-width

Was inspired by the many *"directed path"* problems.

K-width is the *maximum number* of distinct simple *s-t paths* over all $s, t \in V(G)$

Some features

- at least as big as DAG-width, incomparable with cycle-rank
- can be arbitrarily large on DAGs
- easily computed in time k.poly($|V(G)|$)

“Path” problems

HAM	–	Hamiltonian path
k-PATH	–	given k pairs (s_i, t_i) of vertices, find k vertex-disjoint paths between those pairs
c-PATH	–	as above, but $c \geq 2$ is fixed

Problem	K-width	DAG-depth	DAG-width	Cycle-rank	DAG	Bi-rank-width
HAM	**FPT**	**FPT**	XP * W[2]-hard	XP *	P	XP W[2]-hard
c-PATH	**FPT**	**FPT**	XP *	XP *	P	**FPT**
k-PATH	**p-NPC**	**p-NPC**	**NPC**	**NPC**	**NPC**	*open*

Oriented colouring

- The *chromatic number* $\chi(G)$ of G is a minimal c such that G has a homomorphism into K_c.
- The *oriented chromatic number* $\chi_o(G)$ of G is a minimal c such that G has a homomorphism into *some* orientation of K_c.
- Implies that between any two vertices of any two given colours the edges must go in the same direction ($\chi_o(C_5) = 5$).
- The number of colours c is fixed (i.e. not part of the input).

Problem	K-width	DAG-depth	DAG-width	Cycle-rank	DAG	Bi-rank-width
c-OCN	**p-NPC**	**p-NPC**	NPC	NPC	NPC	**FPT**

Directed Feedback Vertex Set and Kernel

DFVS

- We look for a minimum cardinality set $S \subset V(G)$ such that $G \setminus S$ is acyclic.
- We consider the optimization variant (it's FPT with $k = |S|$).

Kernel

- An independent set $S \subseteq V(G)$ such that for every $x \in V(G) \setminus S$ there is an arc from x into S.

Problem	K-width	DAG-depth	DAG-width	Cycle-rank	DAG	Bi-rank-width
DFVS	*open*	*open*	p-NPC	p-NPC	P	**FPT**
KERNEL	p-NPC	p-NPC	p-NPC	p-NPC	P	**FPT**

Maximum Directed Cut

- The problem is to partition $V(G)$ into V_0 and V_1 such that the cardinality of $\{(u, v) \in E(G) \mid u \in V_0, v \in V_1\}$ is maximized.
- Often stated with edge weights.

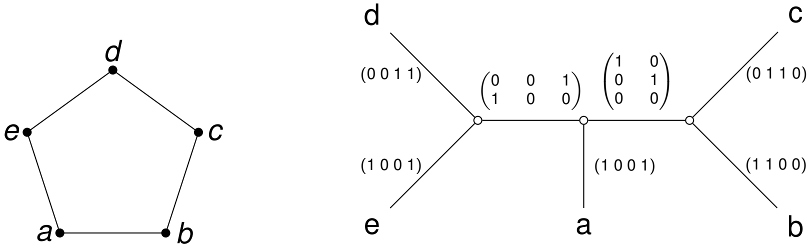
Problem	K-width	DAG-depth	DAG-width	Cycle-rank	DAG	Bi-rank-width
MAXDICUT	p-NPC	p-NPC	NPC	NPC	NPC	**XP**

The results

Problem	K-width	DAG-depth	DAG-width	Cycle-rank	DAG	Bi-rank-width
HAM	**FPT**	**FPT**	XP * W[2]-hard	XP *	P	XP W[2]-hard
c-PATH	**FPT**	**FPT**	XP *	XP *	P	**FPT**
k-PATH	**p-NPC**	**p-NPC**	**NPC**	**NPC**	**NPC**	*open*
DIDS	**p-NPC**	**p-NPC**	**NPC**	**NPC**	**NPC**	**FPT**
DISTP	**p-NPC**	**p-NPC**	**NPC**	**NPC**	**NPC**	**FPT**
MAXDICUT	p-NPC	p-NPC	NPC	NPC	NPC	**XP**
c-OCN	**p-NPC**	**p-NPC**	NPC	NPC	NPC	**FPT**
DFVS	*open*	*open*	p-NPC	p-NPC	P	**FPT**
KERNEL	p-NPC	p-NPC	p-NPC	p-NPC	P	**FPT**
φ-MSO_1MC	**p-NPH**	**p-NPH**	NPH	NPH	NPH	**FPT**
φ-LTLMC	**p-coNPH**	**p-coNPH**	**coNPH**	**coNPH**	**coNPC**	**p-coNPH**
PARITY	XP	XP	XP *	XP *	P	XP

Rank-Width

- Defined by Oum and Seymour, 2004.
- Basically clique-width done right.



- φ-MSO_1MC is FPT.
- Better representation for algorithms: parse trees.

Bi-Rank-Width

- Straightforward generalization of rankwidth.
- Instead of one matrix we use two matrices, one for each direction.
- Defined by Kanté, 2008.
- Parse-trees generalize as well.
- φ-MSO_1MC is FPT.
- For each fixed $t \in \mathbb{N}$ there is an algorithm which, given a graph G, in time $\mathcal{O}(n^3)$ either outputs a bi-rank-decomposition of G of width at most t, or certifies that such decomposition does not exist.

Thank you

Thank you for your attention