Finding Sup-Transition-Minors with SAT

Benedikt Klocker

Algorithms and Complexity Group
Institute of Logic and Computation
TU Wien

PHD Seminar, 5 November 2018

A Model for EBSTM

Given transitioned graphs $(G, \mathcal{T})$ and $(H, \mathcal{S})$. The model consists of

1. a partial surjective function $\varphi : V(G) \nrightarrow V(H)$,
2. a partial injective and surjective function $\kappa : E(G) \nrightarrow E(H)$,
3. a partial injective function $\theta : E(G) \nrightarrow V(H)$,
4. for each $w \in V(H)$ a pair (T_w, S_w) of transitions with $T_w \in \mathcal{T}$ and $S_w \in \mathcal{S}(w)$.
5. for each $w \in V(H)$ two simple trees C_w^1 and C_w^2 with $V(C_w^i) \subseteq V(G)$ for $i = 1, 2$.

$$
\begin{aligned}
&E(C_w^i) \subseteq r_G[E(G)] && \forall w \in V(H), \forall i = 1,2 \\
&\kappa(e) = f \Rightarrow \varphi[r_G(e)] = r_H(f) && \forall e \in E(G), \forall f \in E(H) \\
&V(C_w^1) \cup V(C_w^2) = \varphi^{-1}(w) && \forall w \in V(H) \\
&\{\pi_1(T_w)\} = V(C_w^1) \cap V(C_w^2) && \forall w \in V(H) \\
&\pi_2(T_w) \subseteq \kappa^{-1}[\pi_2(S_w)] \cup \theta^{-1}[w] \cup E_w^1 && \forall w \in V(H) \\
&\left(\kappa^{-1}[\pi_2(S_w)] \cap E(\pi_1(T_w))\right) \cup \theta^{-1}[w] \subseteq \pi_2(T_w) && \forall w \in V(H) \\
&e \in \mathrm{dom}(\kappa) \wedge \kappa(e) \in \pi_2(S_w) \Rightarrow r_G(e) \cap V(C_w^1) \neq \emptyset && \forall w \in V(H), \forall e \in E(G) \\
&e \in \mathrm{dom}(\kappa) \wedge \kappa(e) \in E(w) \setminus \pi_2(S_w) \Rightarrow r_G(e) \cap V(C_w^2) \neq \emptyset && \forall w \in V(H), \forall e \in E(G) \\
&v \in V(C_w^1) \setminus \{\pi_1(T_w)\} \wedge \deg_{C_w^1}(v) = 1 \wedge v \notin \bigcup r_G[\theta^{-1}[w]] && \forall w \in V(H), \forall v \in V(G) \\
&\quad \Rightarrow E(v) \cap \kappa^{-1}[\pi_2(S_w)] \neq \emptyset && \\
&E_{C_w^1}(\pi_1(T_w)) \subseteq r_G[\pi_2(T_w)] && \forall w \in V(H) \\
&\theta(e) = w \Rightarrow r_G(e) \subseteq V(C_w^1) && \forall e \in E(G), \forall w \in V(H) \\
&\theta(e) = w \Rightarrow r_G(e) \notin E(C_w^1) && \forall e \in E(G), \forall w \in V(H)
\end{aligned}
$$

Let $f : A \nrightarrow B$ be a partial function. Use binary variables $x_{a,b}$ for $a \in A$ and $b \in B$: $x_{a,b}$ is true if and only if $f(a) = b$. Assuring that x represents a partial function:

$$partial(x) := \neg(x_{a,b_1} \wedge \neg x_{a,b_2}) \quad \forall a \in A, \{b_1, b_2\} \subseteq B$$

Number of clauses:

$$|partial(x)| := |A| \cdot \binom{|B|}{2}$$

Assuring that x represents a function:

$$function(x) := partial(x) \wedge (\bigvee_{b \in B} x_{a,b} \quad \forall a \in A)$$

$$|function(x)| = |A| \cdot \binom{|B|}{2} + |A|$$

Let x represent a (partial) function as described before.

$$injective(x) := \neg(x_{a_1,b} \wedge x_{a_2,b}) \quad \forall \{a_1, a_2\} \subseteq A, b \in B$$

$$|injective(x)| = \binom{|A|}{2}|B|$$

$$surjective(x) := \bigvee_{a \in A} x_{a,b} \quad \forall b \in B$$

$$|surjective(x)| = |B|$$

Let G be a *simple* undirected graph. To model a subtree of G we use the directed version of G by replacing each edge with two opposite arcs.

Variables:

- $(r_v)_{v \in V(G)}$ true iff v is the root vertex of the out-tree
- $(x_v)_{v \in V(G)}$ decides if the vertex v is in the subtree
- $(y_a)_{a \in A(G)}$ decides if the arc a is in the subtree
- $(t_{v_1,v_2})_{v_1,v_2 \in V(G)}$ the transitive closure of the arcs in the tree

SAT - Modelling Trees cont.

$tree(r, x, y, t) :=$

$$
\begin{array}{llr}
 & \neg(r_{v_1} \wedge r_{v_2}) & \forall\{v_1, v_2\} \subseteq V(G) \\
\wedge & r_v \rightarrow x_v & \forall v \in V(G) \\
\wedge & y_{(v_1,v_2)} \rightarrow (x_{v_1} \wedge x_{v_2}) & \forall(v_1, v_2) \in A(G) \\
\wedge & \neg(y_{(v_1,v)} \wedge y_{(v_2,v)}) & \forall v \in V, \{v_1, v_2\} \subseteq N(v) \\
\wedge & x_v \rightarrow \left(\bigvee_{v_1 \in N(v)} y_{(v_1,v)} \vee r_v \right) & \forall v \in V(G) \\
\wedge & y_{(v_1,v_2)} \rightarrow t_{v_1,v_2} & \forall(v_1, v_2) \in A(G) \\
\wedge & t_{v_1,v_2} \wedge t_{v_2,v_3} \rightarrow t_{v_1,v_3} & \forall v_1, v_2, v_3 \in V(G) \\
\wedge & \neg t_{v,v} & \forall v \in V
\end{array}
$$

$$|tree(r, x, y, t)| \leq \binom{|V|}{2} + 3|V| + 6|E| + \frac{1}{2}|E|(\Delta(G) - 1) + |V|^3$$

The SAT Model - Variables

- $x_{v,w}$ … partial surjective function $\varphi : V(G) \nrightarrow V(H)$.
- $y_{e,f}$ … partial injective surjective function $\kappa : E(G) \nrightarrow E(H)$.
- $z_{e,w}$ … partial injective function $\theta : E(G) \nrightarrow V(H)$.
- $a_{w,T}$ … injective function representing $T_w = T$.
- $b_{w,S}$ … injective function representing $S_w = S$ with restriction $S \in \mathcal{S}(w)$.
- $o_{v,w,i}$ … vertex indicator for subtree C^i_w.
- $p_{a,w,i}$ … arc indicator for subtree C^i_w.
- t_{v_1,v_2} … transitive closure for arcs in all trees together.

  Number of variables:

$$|V(G)||V(H)| + |E(G)||E(H)| + |E(G)||V(H)| + |V(H)||\mathcal{T}| + |\mathcal{S}| + 2|V(G)||V(H)| + 4|E(G)||V(H)| + |V|^2$$

The SAT Model - Constraints

Base-Model:

$$\kappa(e) = f \Rightarrow \varphi[r_G(e)] = r_H(f) \quad \forall e \in E(G), \forall f \in E(H)$$

SAT-Model:

$$\forall e = v_1 v_2 \in E(G), \forall f = w_1 w_2 \in E(H)$$

$$y_{e,f} \rightarrow ((x_{v_1,w_1} \wedge y_{v_2,w_2}) \vee (x_{v_2,w_1} \wedge y_{v_1,w_2}))$$

Number of clauses:

$$4|E(G)||E(H)|$$

The SAT Model - Constraints 2

Base-Model:

$$V(C_w^1) \cup V(C_w^2) = \varphi^{-1}(w) \quad \forall w \in V(H)$$

SAT-Model:

$$\forall v \in V(G), \forall w \in V(H)$$

$$(o_{v,w,1} \vee o_{v,w,2}) \leftrightarrow x_{v,w}$$

Number of clauses:

$$3|V(G)||V(H)|$$

The SAT Model - Constraints 3

Base-Model:

$$\{\pi_1(T_w)\} = V(C_w^1) \cap V(C_w^2) \quad \forall w \in V(H)$$

SAT-Model:

$$\forall v \in V(G), \forall w \in V(H)$$

$$\left(\bigvee_{T \in \mathcal{T}(v)} a_{w,T} \right) \leftrightarrow (o_{v,w,1} \wedge o_{v,w,2})$$

Number of clauses:

$$\leq (1 + \Delta(G))|V(G)||V(H)|$$

The SAT Model - Constraints 4

Base-Model:

$$\pi_2(T_w) \subseteq \kappa^{-1}[\pi_2(S_w)] \cup \theta^{-1}[w] \cup E_w^1 \quad \forall w \in V(H)$$

SAT-Model:

$$\forall e = v_1 v_2 \in E(G), \forall w \in V(H)$$

$$\left(\bigvee_{T \in \mathcal{T}: e \in \pi_2(T)} a_{w,T} \right) \rightarrow \bigvee_{S \in \mathcal{S}(w)} \left(b_{w,S} \wedge \bigvee_{f \in \pi_2(S)} y_{e,f} \right) \vee z_{e,w} \vee p_{(v_1,v_2),w,1} \vee p_{(v_1,v_2),w,2}$$

Number of clauses:

$$\leq 8|E(G)||V(H)|$$

The SAT Model - Constraints 5

Base-Model:

$$\left(\kappa^{-1}[\pi_2(S_w)] \cap E(\pi_1(T_w))\right) \cup \theta^{-1}[w] \subseteq \pi_2(T_w) \quad \forall w \in V(H)$$

SAT-Model:

$$\forall w \in V(H), \forall S \in \mathcal{S}(w), \forall T \in \mathcal{T}, \forall e \in E(\pi_1(T)) \setminus \pi_2(T)$$

$$a_{w,T} \wedge b_{w,S} \rightarrow \neg \bigvee_{f \in \pi_2(S)} y_{e,f}$$

$$\forall w \in V(H), T \in \mathcal{T}, e \in E(G) \setminus \pi_2(T)$$

$$a_{w,T} \rightarrow \neg z_{e,w}$$

Number of clauses:

$$\leq 2|\mathcal{S}||\mathcal{T}|(\Delta(G) - 2) + |V(H)||\mathcal{T}|(|E| - 2)$$

The SAT Model - Constraints 6

Base-Model:

$$\forall w \in V(H), \forall e \in E(G)$$

$$e \in \mathrm{dom}(\kappa) \wedge \kappa(e) \in \pi_2(S_w) \Rightarrow r_G(e) \cap V(C_w^1) \neq \emptyset$$

SAT-Model:

$$\forall w \in V(H), \forall S \in \mathcal{S}(w), \forall e = v_1 v_2 \in E(G)$$

$$\left(b_{w,S} \wedge \bigvee_{f \in \pi_2(S)} y_{e,f} \right) \rightarrow (o_{v_1,w,1} \vee o_{v_2,w,1})$$

Number of clauses:

$$2|\mathcal{S}||E(G)|$$

The SAT Model - Constraints 7

Base-Model:

$$\forall w \in V(H), \forall e \in E(G)$$

$$e \in \mathrm{dom}(\kappa) \wedge \kappa(e) \in E(w) \setminus \pi_2(S_w) \Rightarrow r_G(e) \cap V(C_w^2) \neq \emptyset$$

SAT-Model:

$$\forall w \in V(H), \forall S \in \mathcal{S}(w), \forall e = v_1 v_2 \in E(G)$$

$$\left(b_{w,S} \wedge \bigvee_{f \in E(w) \setminus \pi_2(S)} y_{e,f} \right) \rightarrow (o_{v_1,w,2} \vee o_{v_2,w,2})$$

Number of clauses:

$$2|\mathcal{S}||E(G)|$$

The SAT Model - Constraints 8

Base-Model:

$$\forall w \in V(H), \forall v \in V(G)$$

$$v \in V(C_w^1) \setminus \{\pi_1(T_w)\} \wedge \deg_{C_w^1}(v) = 1 \wedge v \notin \bigcup r_G[\theta^{-1}[w]]$$
$$\Rightarrow E(v) \cap \kappa^{-1}[\pi_2(S_w)] \neq \emptyset$$

SAT-Model:

$$\forall w \in V(H), \forall S \in \mathcal{S}(w), \forall v \in V(G)$$

$$(b_{w,S} \wedge o_{v,w,1} \bigwedge_{v' \in N(v)} \neg p_{(v,v'),w,1} \wedge \bigwedge_{e \in E(v)} \neg z_{e,w}) \rightarrow$$
$$\left(\bigvee_{e \in E(v), f \in \pi_2(S)} y_{e,f} \vee o_{v,w,2} \right)$$

Number of clauses: $|\mathcal{S}||V(G)|$

The SAT Model - Constraints 9

Base-Model:

$$E_{C_w^1}(\pi_1(T_w)) \subseteq r_G[\pi_2(T_w)] \quad \forall w \in V(H)$$

SAT-Model:

$$\forall w \in V(H), \forall T \in \mathcal{T}, \forall v \in N(\pi_1(T)) \setminus \bigcup r_G[\pi_2(T)]$$

$$a_{w,T} \rightarrow \neg p_{(\pi_1(T),v),w,1} \wedge \neg p_{(v,\pi_1(T)),w,1}$$

Number of clauses:

$$\leq 2|V(H)||\mathcal{T}||\Delta(G) - 2|$$

The SAT Model - Constraints 10

Base-Model:

$$\theta(e) = w \Rightarrow r_G(e) \subseteq V(C_w^1) \quad \forall e \in E(G), \forall w \in V(H)$$

SAT-Model:

$$\forall w \in V(H), \forall e = v_1 v_2 \in E(G)$$

$$z_{e,w} \rightarrow (o_{v_1,w,1} \wedge o_{v_2,w,1})$$

Number of clauses:

$$2|V(H)||E(G)|$$

Base-Model:

$$\theta(e) = w \Rightarrow r_G(e) \notin E(C_w^1) \quad \forall e \in E(G), \forall w \in V(H)$$

SAT-Model:

$$\forall w \in V(H), \forall e = v_1 v_2 \in E(G)$$

$$z_{e,w} \rightarrow (\neg p_{(v_1,v_2),w,1} \wedge \neg p_{(v_2,v_1),w,1})$$

Number of clauses:

$$2|V(H)||E(G)|$$

Compare SAT with MIP approach

- Constructing the SAT model with Python to have a fair comparison with the MIP model
- Solving the SAT model with Glucose (a SAT solver written in C)
- Three instance sets
  - S1: contracting three random perfect matchings for all snarks with up to 26 vertices and 1000 snarks with 28 vertices
  - S2: contracting all perfect pseudo-matchings for all snarks with up to 22 vertices (removing duplicate instances by automorphism check)
  - G1: randomly generated 4-regular completely transitioned graphs (for G and H)

Compare SAT with MIP approach - S1

		MIP				SAT		
$\|V\|$	instances	t[s]	inf	feas	tl	t[s]	inf	feas
10	4	0.17	0	4	0	0.11	0	4
18	8	6.45	0	8	0	0.20	0	8
20	24	3.96	0	24	0	0.26	0	24
22	124	12.34	2	121	1	0.31	3	121
24	620	14.98	12	604	4	0.36	16	604
26	5188	20.53	23	5124	41	0.41	64	5124
28	4004	34.32	12	3973	19	0.46	31	3973

Compare SAT with MIP approach - S2

		MIP				SAT		
$\|V\|$	instances	t[s]	inf	feas	tl	t[s]	inf	feas
18	98	183.63	83	15	0	0.24	83	15
20	1116	251.36	700	416	0	0.31	700	416
22	10694	349.24	5813	4873	8	0.38	5821	4873

Compare SAT with MIP approach - G1

			MIP				SAT		
$\lvert V(G)\rvert$	$\lvert V(H)\rvert$	instances	t[s]	inf	feas	tl	t[s]	inf	feas
09	5	30	227.53	15	15	0	0.48	15	15
09	6	30	3387.63	26	4	0	0.54	26	4
09	7	30	7959.24	30	0	0	0.60	30	0
10	5	30	208.88	11	19	0	0.57	11	19
10	6	30	7244.40	26	4	0	1.29	26	4
10	7	30	32582.47	22	0	8	1.48	30	0
11	5	30	146.22	5	25	0	0.41	5	25
11	6	30	15001.70	14	9	7	3.16	21	9
11	7	30	43200.00	6	1	23	3.14	29	1
12	5	30	110.25	2	28	0	0.52	2	28
12	6	30	1593.95	1	21	8	3.14	9	21
12	7	30	43200.00	0	1	29	6.03	29	1
13	5	30	114.56	0	28	2	0.58	2	28
13	6	30	2189.21	0	20	10	4.10	10	20
13	7	30	43200.00	0	3	27	14.38	23	7
14	5	30	41.97	0	30	0	0.47	0	30
14	6	30	748.52	0	27	3	2.37	2	28
14	7	30	43200.00	0	5	25	26.56	22	8
15	5	30	42.91	0	30	0	0.53	0	30
15	6	30	655.60	0	28	2	1.93	1	29
15	7	30	43200.00	0	14	16	19.10	12	18

A "real" world application ;)

Circuit Double Cover Conjecture (CDCC)

Let G be a bridgeless undirected graph. Then, there exists a collection of circuits of G, such that each edge is contained in exactly two circuits.

Theorem (Jaeger(1985))

Every minimal counter example to the CDCC must be a snark.

A "real" world application ;)

Definition (Compatible Circuit Decomposition)

Let $(G, \mathcal{T})$ be a transitioned graph. A *compatible circuit decomposition of* G is a circuit decomposition $\mathcal{C}$ of G such that for all transitions in $\mathcal{T}$ there is no circuit in $\mathcal{C}$ which contains both edges of the transition.

Theorem

If a snark G contains a perfect (pseudo-)matching such that its contraction leads to a transitioned graph for which there exists a CCD, then there exists a CDC for G.

A "real" world application ;)

Theorem (Fleischner(1980))

If a transitioned graph is planar it has a CCD.

Theorem (Fhan and Zhang(2000))

If a transitioned graph is K_5-minor-free it has a CCD.

Theorem (Fleischner et al.(2018))

If a transitioned graph is bad-K_5-minor-free it has a CCD.

A "real" world application ;)

We want to check for a graph if it contains a planarizing/K_5-minor-free/bad-K_5-minor-free/CCD-containing perfect pseudo-matching:

1. Given a snark G as input
2. Generate all perfect pseudo matchings and the corresponding contracted transitioned graphs
3. Check for all contracted graphs if they are planar, if we find one *stop*
4. Check for all contracted graphs if one of them is K_5-minor-free, if we find one *stop*
5. Check for all contracted graphs if one of them is bad-K_5-minor-free, if we find one *stop*
6. Check for all contracted graphs if one of them contains a CCD, if we find one *stop*

Implementation Details

- Framework is implemented in C++ (due to better performance compared to Python)
- Planarity is checked with boosts implementation of the Boyer-Myrvold planarity test (linear time)
- K_5-minor-freeness is checked at the moment with a SAT-model (although it could be checked in linear time with a quite complex algorithm) using Glucose as SAT-solver
- bad-K_5-minor freeness is checked with the above presented SAT-model using Glucose as SAT solver
- CCD-containment is checked with a simple SAT-model using Glucose as SAT-solver

Could solve all snarks with up to 32 vertices. (1 918 812 graphs)

- 1 893 564 graphs contain a planarizing perfect pseudo-matching
- 6 118 graphs contain no planarizing perfect pseudo-matching but a K_5-minor-free perfect pseudo-matching
- 19 130 graphs contain no K_5-minor-free perfect pseudo-matching but a bad-K_5-minor-free perfect pseudo matching

All snarks with up to 32 vertices contain a bad-K_5-minor-free perfect pseudo matching.

Note: most runtime is used in checking K_5-minor-freeness.
Bad-K_5-minor-freeness is proven almost always with the first tested perfect-pseudo-matching.

Results - Skipping K_5-minor tests

Could solve all snarks with up to 34 vertices. (27 205 765 graphs)

- 26 298 275 graphs contain a planarizing perfect pseudo-matching
- 907 490 graphs contain no planarizing perfect pseudo-matching but a bad-K_5-minor-free perfect pseudo-matching

All snarks with up to 34 vertices contain a bad-K_5-minor-free perfect pseudo matching.

Results - Checking all perfect pseudo-matchings

- Was able to check all perfect pseudo-matchings for all snarks with up to 26 vertices

Future Work

- Check "all" known snarks with up to 40 vertices
- Implement an efficient K_5-minor-check
- Symmetry breaking in perfect pseudo-matching construction
- Symmetry breaking in bad-K_5-minor SAT
- Compare the SAT approach with a CP approach