

# SAT Problems and Bounded Rank-width

R. Ganian, P. Hliněný, *J. Obdržálek*

Masaryk University, Brno

FSTTCS, Chennai, December 15, 2010

## SAT

- satisfiability for Boolean formulas in CNF
- NP-complete

## #SAT

- aka propositional model counting
- compute the number of satisfying assignments
- #P-hard

## MAX-SAT

- compute the maximum number of simultaneously satisfiable clauses
- already NP-hard to approximate within some constant

## SAT

- satisfiability for Boolean formulas in CNF
- NP-complete

## #SAT

- aka propositional model counting
- compute the number of satisfying assignments
- #P-hard

## MAX-SAT

- compute the maximum number of simultaneously satisfiable clauses
- already NP-hard to approximate within some constant

## SAT

- satisfiability for Boolean formulas in CNF
- NP-complete

## #SAT

- aka propositional model counting
- compute the number of satisfying assignments
- #P-hard

## MAX-SAT

- compute the maximum number of simultaneously satisfiable clauses
- already NP-hard to approximate within some constant

## size of the solution

- works for MAX-SAT [CK04,RO'S09]
- not suitable for #SAT

## formula as a graph

### PICTURE

- now we can chose our favourite decomposition
- tree-width and clique-width were examined [FMR08,GP08,SS10]

# Where do we stand?

## Fischer, Makowsky, Ravve 2008

- FPT algorithm for #SAT and clique-width (and tree-width)
- *singly exponential* in clique-width

## Samer and Szeider, 2008

A single-exponential algorithm (for #SAT) is due to Fisher, Makowsky, and Ravve [FMR08]. However, both algorithms rely on clique-width approximation algorithms. The known polynomial-time algorithms for that purpose admit an exponential approximation error [HO08] and are of limited practical value.

- rank-width can be exponentially smaller than clique-width
- FPT algorithm for computing rank-width
- straightforward translation of [FMR08] would give us *double exponential* runtime

# Where do we stand?

## Fischer, Makowsky, Ravve 2008

- FPT algorithm for #SAT and clique-width (and tree-width)
- *singly exponential* in clique-width

## Samer and Szeider, 2008

A single-exponential algorithm (for #SAT) is due to Fisher, Makowsky, and Ravve [FMR08]. However, both algorithms rely on clique-width approximation algorithms. The known polynomial-time algorithms for that purpose admit an exponential approximation error [HO08] and are of limited practical value.

- rank-width can be exponentially smaller than clique-width
- FPT algorithm for computing rank-width
- straightforward translation of [FMR08] would give us *double exponential* runtime

# Where do we stand?

## Fischer, Makowsky, Ravve 2008

- FPT algorithm for #SAT and clique-width (and tree-width)
- *singly exponential* in clique-width

## Samer and Szeider, 2008

A single-exponential algorithm (for #SAT) is due to Fisher, Makowsky, and Ravve [FMR08]. However, both algorithms rely on clique-width approximation algorithms. The known polynomial-time algorithms for that purpose admit an exponential approximation error [HO08] and are of limited practical value.

- rank-width can be exponentially smaller than clique-width
- FPT algorithm for computing rank-width
- straightforward translation of [FMR08] would give us *double exponential* runtime

## Theorem

*#SAT and MAX-SAT have FPT algorithms running in time*

$$\mathcal{O}(t^3 \cdot 2^{3t(t+1)/2} \cdot |\varphi|),$$

*if a rank-decomposition of width  $t$  is given, and*

$$\mathcal{O}(2^{\Theta(t^2)} \cdot |\varphi|^3)$$

*otherwise, where  $t$  is the rank-width of the input formula  $\varphi$ .*

- presents an exponential time improvement on some instances, even when the optimal clique decomposition is given
- no large hidden constants in the  $\mathcal{O}$  notation
- class of graphs of low rank-width is very rich (i.e. distance hereditary graphs have rank-width 1)

## Theorem

#SAT and MAX-SAT have FPT algorithms running in time

$$\mathcal{O}(t^3 \cdot 2^{3t(t+1)/2} \cdot |\varphi|),$$

if a rank-decomposition of width  $t$  is given, and

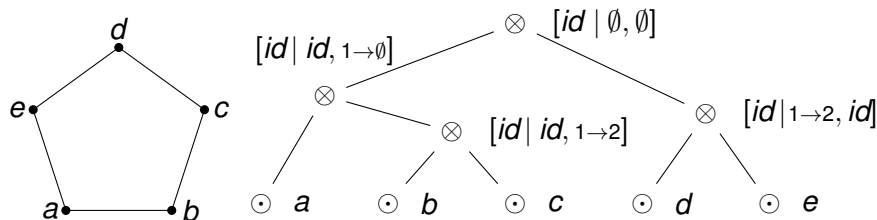
$$\mathcal{O}(2^{\Theta(t^2)} \cdot |\varphi|^3)$$

otherwise, where  $t$  is the rank-width of the input formula  $\varphi$ .

- presents an exponential time improvement on some instances, even when the optimal clique decomposition is given
- no large hidden constants in the  $\mathcal{O}$  notation
- class of graphs of low rank-width is very rich (i.e. distance hereditary graphs have rank-width 1)



# Parse trees for rank-width



- *labelling*  $lab : V(G) \rightarrow 2^{\{1,2,\dots,t\}}$  (multicolouring)
- *leaves* labelled  $\{1\}$
- *joins* – composition operator  $\otimes [g \mid f_1, f_2]$ 
  - $g$  – labels to connect
  - $f_1, f_2$  – relabelling of the left (right) subgraph *after*

Instead of directed graphs, we use the following notion:

## Definition

*Signed graph*  $G$  is a graph with two distinct edge sets,  $E^+(G)$  and  $E^-(G)$ , which induce the subgraphs  $G^+$  and  $G^-$ .

PICTURE AGAIN

## modified parse trees

- with  $G^+$  ( $G^-$ ) we associate labelling  $lab^+$  ( $lab^-$ )
- parse trees pair  $T = (T^+, T^-)$  s.t.:
  - $T^+$  generates  $G^+$  and  $T^-$  generates  $G^-$
  - the underlying rooted trees  $T^+$  and  $T^-$  are identical

Instead of directed graphs, we use the following notion:

## Definition

*Signed graph*  $G$  is a graph with two distinct edge sets,  $E^+(G)$  and  $E^-(G)$ , which induce the subgraphs  $G^+$  and  $G^-$ .

PICTURE AGAIN

## modified parse trees

- with  $G^+$  ( $G^-$ ) we associate labelling  $lab^+$  ( $lab^-$ )
- parse trees pair  $T = (T^+, T^-)$  s.t.:
  - $T^+$  generates  $G^+$  and  $T^-$  generates  $G^-$
  - the underlying rooted trees  $T^+$  and  $T^-$  are identical

- by dynamic programming on a parse tree pair
- *partial assignment* for a node  $F_i$ : mapping  $\nu : \text{Var}_i \rightarrow \{0, 1\}$

## Tables

For a node  $F_1$  and  $\nu_1 : C_1 \rightarrow \{0, 1\}$  we need to know

- I. the true literals in  $F_1$  (w.r.t.  $\nu_1$ ) which are available to satisfy the clauses of  $F_\varphi$
- II. that for every clause  $c$  in  $F_1$  at least one of the following is true
  - $c$  is satisfied by a literal from  $F_1$
  - $c$  is expected to be satisfied by a literal from  $F_\varphi \setminus F_1$

- II. that for every clause  $c$  in  $F_1$  at least one of the following is true
- $c$  is satisfied by a literal from  $F_1$
  - $c$  is expected to be satisfied by a literal from  $F_\varphi \setminus F_1$

## What are the expectations?

- introduced by Bui-Xuan, Telle and Vatshelle (2010)
- in addition to the information about the partial solution for the part of the graph processed so far, we also keep the information what will be required from the complimentary partial solution for the unprocessed part of the graph
- although looking expensive in space, it may *greatly reduce* the description of the partial solution

# Assignment shapes

For a node  $F_1$  of the parse tree a partial solution  $\nu_1 : V(F_1) \rightarrow \{0, 1\}$  is the *assignment of shape*  $(\Sigma^+, \Sigma^-, \Pi^+, \Pi^-)$  if

- I.  $\Sigma^+(\Sigma^-)$  is the subspace of  $GF(2)^t$  generated by the set of labeling vectors  $lab^+(\nu_1^{-1}(1))(lab^-(\nu_1^{-1}(0)))$
- II.  $\Pi^+, \Pi^-$  (the *expectation part*) are subspaces of  $GF(2)^t$  such that, for every clause  $c \in V(F_1) \cap C$ , at least one of the following is true
  - $c$  is adjacent to some vertex from  $\nu_1^{-1}(1)$  in  $F_1^+$  or to one from  $\nu_1^{-1}(0)$  in  $F_1^-$ , or
  - the label vector  $lab^+(c)$  is not orthogonal to  $\Pi^+$  or  $lab^-(c)$  is not orthogonal to  $\Pi^-$

## Lemma

*Assume  $t$ -labeled graphs  $\bar{G}$  and  $\bar{H}$ , and arbitrary  $X \subseteq V(\bar{G})$  and  $y \in V(\bar{H})$ . In the join graph  $\bar{G} \otimes \bar{H}$ , the vertex  $y$  is adjacent to some vertex in  $X$  if and only if the vector subspace spanned by the  $\bar{G}$ -labelings of the vertices of  $X$  is not orthogonal to the  $\bar{H}$ -labeling vector of  $y$  in  $\text{GF}(2)^t$ .*

## Lemma

*The number of subspaces  $S(t)$  of  $GF(2)^t$  satisfies  $S(t) \leq 2^{t(t+1)/4}$  for all  $t \geq 12$*

- time dominated by

# Conclusions