Exact Max 2-SAT: Easier and Faster

Martin Fürer
Shiva Prasad Kasiviswanathan
Pennsylvania State University, U.S.A

MAX 2-SAT

- Input: A 2-CNF fomula F with weights on clauses.
- Good assignment is one that maximizes the sum of weights of satisfied clauses.
- Optimization Problem: Find a good assignment.
- Counting Problem: Count the number of such good assignments.

An Example

Let $F = (x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1)$.

x_1	x_2	#-clauses satisfied
0	0	3
0	1	3
1	0	3
1	1	2

good assignments

Status of Max 2-SAT

- NP-hard.
- Best approximation ratio known = 0.940.
- APX-hard with inapproximability ratio 21/22 (under $P \neq NP$).
- Better hardness results under Unique Game Conjecture.
- Counting version is #P-complete.

Some Known Results

Authors	Time	Comments
Kojevnikov, Kulikov 2006	$O(2^{m/5.5})$	m = #-clauses.
Scott, Sorkin 2006	$O(2^{19m/100})$	holds for binary-CSP
Williams 2004	$O(2^{wn/3})$	n = #-variables, exponential space

Summary: For polynomial space algorithms with complexity measure n, not much is known.

Our Contributions – I

- Obtain a worst case bound of

$$O(2^{((d(F)-2)/(d(F)-1))n})$$

d(F) = average number of clauses that a variable participates.

- The algorithm uses only polynomial space.
- Same upper bound for both counting and optimization problems.

Our Contributions - II

- We obtain an $O(2^{cn})$ upper bound if the underlying constraint graph has a small separator decomposition.
- Here c is some constant < 1 (i.e., independent of n).

Our Contributions – III

- We introduce a new notion for gadget reductions.

- This notion allows us to obtain same upper bound for problems like
  - Max k-SAT (k constant),
  - Max Cut,
  - Max k-Lin-2 (k constant).

General Idea

- The algorithm uses a DPLL-like recursive decomposition technique.
- The idea is to chose a variable v and to recursively assign v to true and false.

  (a.k.a. we branch on v)
- The aim is to minimize the number of such branchings.

Some Definitions

- Constraint graph: $G(F) = (Var(F), E)$ where
  $Var(F)$ = set of variables of F,
  $E = \{(u,v) \mid u,v$ appear in the same clause of $F\}$.
- Problem 3-SAT:

  Input: 3-CNF formula with weight $w(l)$ on a literal l.

  Good assignment: An assignment M satisfying all the clauses such that

  $$W(M) = \Sigma_{(l \text{ satisfied by } M)} w(l) \text{ is maximized.}$$

Worst case bounds

- Step 1: Parsimonious reduction from Max 2-SAT to 3-SAT.

Clause weighted

Literal weighted

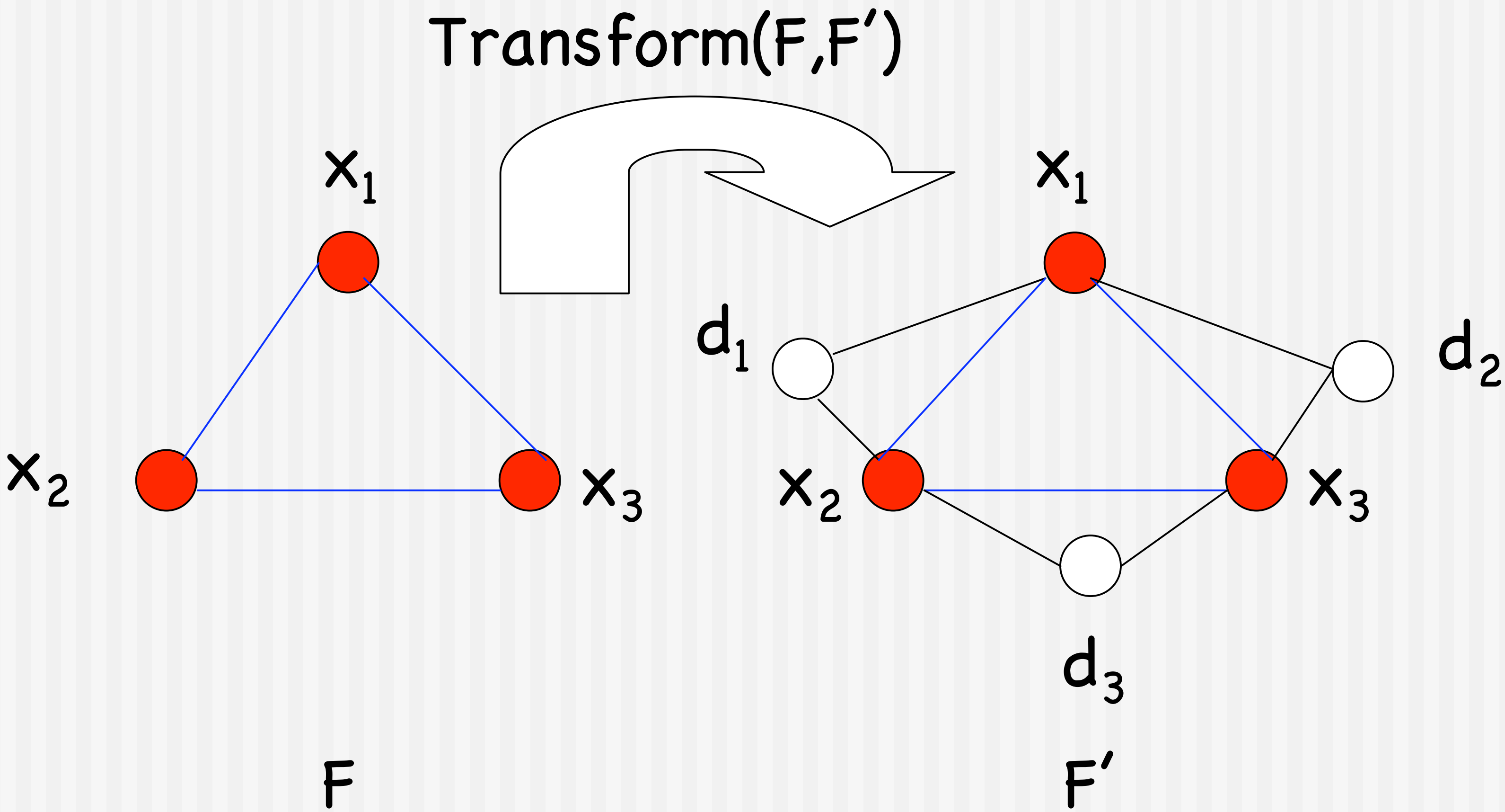
- Step 2: Solve the 3-SAT instance.

Parsimonious Reduction

Function Transform(F,F′)

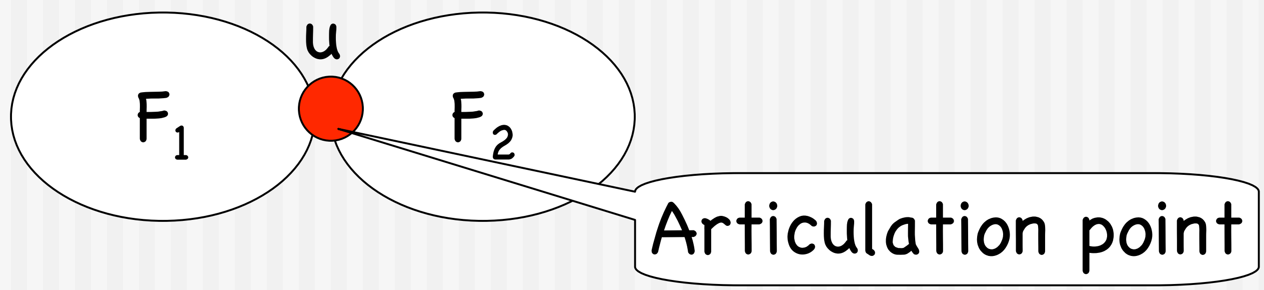
1) For each clause $C = (x_i \vee x_j)$ in F add a clause $(x_i \vee x_j \vee d_C)$ to F′.
2) Assign weights to literals in F′ as:
   - 0 to any literal from $\{x_1,\ldots,x_n,\neg x_1,\ldots,\neg x_n\}$.
   - w(C) (weight of clause C) to literal $\neg d_C$.
   - 0 to literal d_C.

Local Effects of Reduction



An Useful Trick

Let $F = F_1 \wedge F_2$ and F_1, F_2 have exactly one common variable (say u) then one can work with F_1 (or F_2) and use the result to update weight of u.



Advantage: One can solve F_1 and F_2 separately.

First Noted by Dahllöf et al 2005.

Algorithm Local-2-SAT

If there exists an unassigned variable, then

a) pick a variable v with lowest degree in the graph induced by unassigned variables.

b) branch on all but one of v's neighbors simultaneously.

Trick from the previous slide saves one branching.

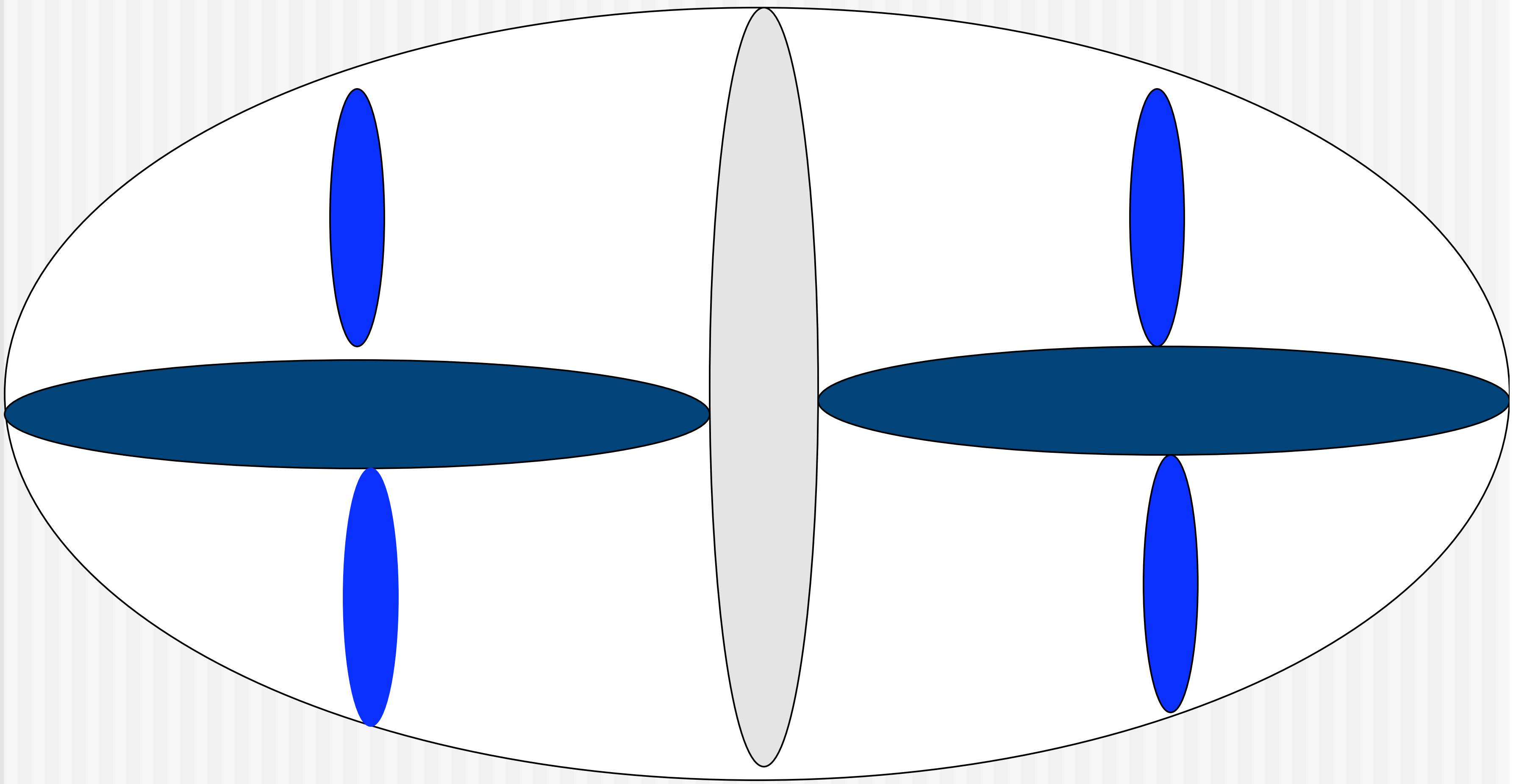
Upper Bound on Running Time

- Theorem 1: Algorithm Local-2-SAT runs in $O(2^{((\Delta(F)-2)/(\Delta(F)-1))n})$ time, where $\Delta(F)$ is the maximum degree in G(F).

    By a more careful analysis.

- Theorem 2: Algorithm Local-2-SAT runs in time $O(2^{((d(F)-2)/(d(F)-1))n})$, where d(F) is the average degree in G(F).

Separator Decomposition



Algorithm Global-2-SAT

1) Recursively find a separator in the graph.
2) Branch on these vertices simultaneously.

Global-2-SAT for Separable Graphs

Theorem 3: If G(F) has a small separator decomposition (i.e., every sub-graph of size k has a separator of size ηk^u with $0<u<1$), then Global-2-SAT runs in

$$O(2^{n^u \eta/(1-\rho^u)}) \text{ time.}$$

A constant < 1 arising out of splitting ratio.

Global-2-SAT worst case bounds

- For many classes of graphs no small separators exist.
- For these graphs we use BFS to obtain a separator.

Theorem 4: Algorithm Global-2-SAT runs in $O(2^{((\Delta(F)-2)/(\Delta(F)-1))n+(\Delta(F)+1)\log n})$ time.

Gadgets

- Gadgets (α-gadgets) introduced by Trevisan et al. defines a reduction from a constraint function to a constraint family.
- We parameterize gadgets by two parameters α, β.
- Advantage: Allows compositions of gadgets.

Definition of (α,β)-gadgets

A (α,β)-gadget for $\alpha, \beta \in \Re^+$, reduces a constraint function $f:\{0,1\}^n \to \{0,1\}$ to a constraint family *H* such that,

a) the result is a finite collection of constraints $\{C_1,\ldots,C_{\beta'}\}$ from *H* over input variables $x_1,\ldots,x_n$ and auxiliary variables $y_1,\ldots,y_m$.

Definition Contd…

b) The weights $\{w_1,\ldots,w_{\beta'}\}$ are assigned such that

$$w_1+w_2+\ldots+w_{\beta'} = \beta.$$

and for Boolean assignments A to $x_1,\ldots,x_n$ and B to $y_1,\ldots,y_m$:

$$(\forall A\colon f(A) = 1)\ \max_B(\textstyle\sum_i w_i C_i(A,B)) = \alpha,$$

$$(\forall B\colon f(A) = 0)\ \max_B(\textstyle\sum_i w_i C_i(A,B)) = \alpha-1.$$

Use of Gadgets

$$\underset{\text{(Opt problem)}}{P} \xrightarrow{(\alpha,\beta)} \underset{\text{(instance F)}}{\text{Max 2-SAT}}$$

Sum of constraint weights in P

Max 2-SAT on F has optimum value αW.
Any solution value S for F corresponds to solution value $S-(\alpha-1)W$ in P.

Parameter β helps us to chain these reductions.

Results with Gadgets

$$\left.\begin{array}{l} f \in H_1 \xrightarrow{(\alpha_1,\beta_1)} H_2 \\ g \in H_2 \xrightarrow{(\alpha_2,\beta_2)} H_3 \end{array}\right\} \Longrightarrow f \in H_1 \xrightarrow{(\beta_1(\alpha_2-1)+\alpha_1,\beta_1\beta_2)} H_3$$

$$\text{Max Cut} \xrightarrow{(2,2)} \text{Max 2-SAT}$$

$$\text{Max k-SAT} \xrightarrow{(3.5(k-2),4(k-2))} \text{Max 2-SAT}$$

Final result: Same upper bound for these problems.

Some Open Problems

- Is there an $O(2^{cn})$ time algorithm for Max 2-SAT?
- Easier problem: Exponential time algorithm for $(1-\varepsilon)$ approximation?

Thank you