Resolution Size by Graph Based Parameters

Ziyi Cai
BASICS Lab, Shanghai Jiao Tong University

Introduction

Preliminaries and Notations

- SAT instances $\phi, \psi, \ldots$.
- Clauses of a SAT $C_1, C_2, \ldots$.
- An undirected *hypergraph* H is a pair $H = (V, E)$ where V is a set of elements called *vertices* and E is a set of non-empty subsets of V called *hyperedges*.

Definition 1.1 (tree decomposition)

Let G be a graph. A *tree decomposition* of G is a tuple $\left(T, (B_t)_{t \in V(T)}\right)$, where T is a tree and B_t the *bag* at t such that the following conditions are satisfied:

- For every $v \in V(G)$ the set
  $$T_v := \{t \in V(T) \mid v \in B_t\}$$
  is nonempty and connected in T, i.e, $T[T_v]$ is a subtree of T.
- For every $e \in E(G)$ there exists a $t \in V(T)$ such that $e \subseteq B_t$.

Treewidth

The *width* of a tree decomposition $\left(T, (B_t)_{t \in V(T)}\right)$ is

$$\operatorname{width}\left(T, (B_t)_{t \in V(T)}\right) := \max\{|B_t| - 1 \mid t \in V(T)\}.$$

The *treewidth* of G is a minimum number of widths of all tree decompositions of G.

Hypergraph of SAT Instances

To each instance ϕ of a SAT, we can associate a hypergraph $H(\phi) = (V, E)$ whose set V of vertices coincides with $\mathrm{Vars}(\phi)$, and whose set E of hyperedges contains for each clause C_i of ϕ, a hyperedge E_i consisting of $\mathrm{Vars}(C_i)$.

Underlying Graphs of Hypergraphs

[HOSG07] discusses some methods to get a graph from a hypergraph.

Primal Graph $V(P(H)) = V(H), E(P(H)) = \bigcup_{e \in E(H)} \{(x, y) \mid x, y \in e\}$

Incidence Graph $V(I(H)) = V(H) \cup E(H), E(I(H)) = \bigcup_{e \in E(H)} \{(x, e) \mid x \in e\}$

We define $P_\phi := P(H(\phi))$ and $I_\phi := I(H(\phi))$ for short.

Theorem 1.2

For any CNF formula ϕ, $tw\left(I_\phi\right) \leq tw\left(P_\phi\right) + 1$.

Example 1.3

Let $\phi = x_1 \vee x_1 \vee \cdots \vee x_m$. Then $\mathrm{tw}\left(P_\phi\right) = m - 1$ while $\mathrm{tw}\left(I_\phi\right) = 1$.

Definition 1.4 (resolution)

Resolution is one of the propositional proof systems and has only one inference rule:

$$\frac{C \vee x \qquad D \vee \bar{x}}{C \vee D}$$

Resolution rules takes two clauses and produces a new implied clause (*resolvent*). A *resolution refutation* of a formula ϕ is a sequence of clauses $C_1, C_2, \ldots, C_k$ such that

- for each $i (1 \leq i \leq k)$, C_i is a clause occurring in ϕ or a resolvent of two previous clauses, and
- the last clause C_k is an empty clause.

The *size* of the refutation is the number k of clauses. The *width* of the refutation is a maximum number of literals in clauses.

Theorem 1.5

Resolution is sound and complete.

Theorem 1.6

If for every unsatisfiable CNF ϕ there exists a polynomially bounded resolution refutation proof, then NP = coNP.

Tree-like Resolution

Tree resolution is a diagramatic form of a resolution refutation. The underlying directed acyclic graph of a tree resolution is a tree. Note that in a tree resolution, initial clauses may be reiterated multiple times, but derived clauses may only be used once.

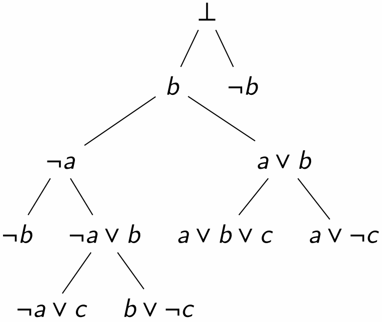

Figure: A tree-like resolution refutation for $(a \vee b \vee c) \wedge (a \vee \neg c) \wedge \neg b \wedge (\neg a \vee c) \wedge (b \vee \neg c)$.

DPLL Algorithm

The **Davis-Putnam-Logemann-Loveland (DPLL) algorithm** [DP60, DLL62] is a search algorithm for deciding the satisfiability of propositional logic formula in conjunctive normal form.

DPLL Algorithm

Algorithm 1 Determine whether a given CNF formula is satisfiable.

function $\mathrm{DPLL}(\phi)$
    while there is a unit clause $\{\ell\}$ in ϕ **do**
        $\phi \leftarrow \textsc{Unit-Propagate}(\ell, \phi)$
    end while
    while there is a literal *l* that occurs pure in ϕ **do**
        $\phi \leftarrow \textsc{Pure-Literal-Assign}(\ell, \phi)$
    end while
    if ϕ is empty **then**
        return true
    end if
    if ϕ contains an empty clause **then**
        return false
    end if
    $l \leftarrow \textsc{Choose-Literal}(\phi)$
    return $\mathrm{DPLL}\left(\phi \cup \{\ell\}\right) \vee \mathrm{DPLL}\left(\phi \cup \left\{\bar{l}\right\}\right)$
end function

Resolution and DPLL-based Algorithms

Theorem 1.7

DPLL and tree-like resolution are polynomially equivalent.

As a consequence, the size of a resolution refutation is a lower bound for time complexity of SAT-solvers.

However, this connection is under assumption of optimality. [AM19] shows that resolution cannot be solved in polynomial time of the resolution size of an input formula unless NP = P holds.

General Intractability of Resolution

Definition 1.8 (pigeonhole principle)

The pigeonhole principle PHP^n_{n-1} is the CNF formula over variables $x_{i,j}$ with $1 \leq i \leq n, 1 \leq j \leq n-1$ with the following clauses:

Pigeon Axioms *Every pigeon hole belong to some hole*: $P_j = \bigvee_{j=1}^{n-1} x_{i,j}$ for $i = 1, \ldots n$.

Hole Axioms *No two pigeons occupy the same hole*: $\overline{x_{i_1,j}} \vee \overline{x_{i_2,j}}$ for $i_1 \neq i_2$ and $j = 1, \ldots, n-1$.

Fact 1.9

PHP^n_{n-1} is not satisfiable.

Theorem 1.10 ([Hak85])

There is a constant $c > 1$ such that any resolution refutation of PHP^n_{n-1} requires size c^n.

Upper Bound Results

Bounded by $\mathrm{tw}(P_\phi)$

Theorem 2.1

For an unsatisfiable formula ϕ with primal treewidth $tw(P_\phi)$, there exists a resolution refutation bounded by $2^{O(tw(P_\phi))} \cdot |\phi|$.

Proof Sketch

A *smooth* tree decomposition is a tree decomposition satisfying that $|B_t \backslash B_{fa(t)}| = 1$ for all $t \in V(T)$ but root. For any graph G there exists a smooth tree decomposition of width $\mathrm{tw}(G)$. We call $B_t \backslash B_{fa(t)}$ the forgotten vertex w.r.t. t.
The desired resolution refutation is obtained by iteratively applying resolutions on forgotten vertices of leaves.

Recall that $\mathrm{tw}(I_\phi) \leq \mathrm{tw}(P_\phi) + 1$.

Conjecture 2.1.1

For an unsatisfiable formula ϕ with incidence treewidth $\mathrm{tw}(I_\phi)$, there exists a resolution refutation bounded by $2^{O(tw(I_\phi))} \cdot |\phi|$.

Bounded by $\mathrm{tw}(I_\phi)$, Under Fixed k

Theorem 2.2

For any k-CNF formula ϕ, $tw(P_\phi) = \Theta(tw(I_\phi))$.

Corollary 2.3

For an unsatisfiable k-CNF formula ϕ with incidence treewidth $tw(I_\phi)$, there exists a resolution refutation bounded by $2^{O(tw(I_\phi))} \cdot |\phi|$.

Bounded by tw(I_ϕ), Under Fixed k

It's well-known that any CNF formula can be converted into an equivalent 3-CNF formula. This conversion can be done without significantly increasing the incidence treewidth.

Theorem 2.4

For any CNF formula ϕ, there exists an equivalent 3-CNF formula ψ such that $tw(I_\psi) = O(tw(I_\phi))$.

Remark 2.4.1

Additional variables are introduced while obtaining the equivalent formula. Hence this result does not necessarily imply we have a resolution refutation with size $2^{O(tw(I_\phi))} \cdot |\phi|$.

Bounded by $\mathrm{pw}(I_\phi)$

Definition 2.5 (path decomposition)

Path decompositions and *pathwidth* are defined similarly as tree decompositions and treewidth, except that in the definition of path decomposition, T is restricted to a simple path.

Fact 2.6

For any graph G, $pw(G) \geq tw(G)$.

Theorem 2.7 ([BT97])

For any graph G, $pw(G) = O(tw(G) \cdot \log|V(G)|)$.

Bounded by $\mathrm{pw}\left(I_\phi\right)$

Theorem 2.8 ([Ima17])

For an unsatisfiable k-CNF formula ϕ with incidence pathwidth $pw\left(I_\phi\right)$, there exists a resolution refutation bounded by $2^{O(pw(I_\phi))} \cdot |\phi|$.

Corollary 2.9

For an unsatisfiable k-CNF formula ϕ with incidence treewidth $tw\left(I_\phi\right)$, there exists a resolution refutation bounded by $|\phi|^{O(tw(I_\phi))}$.

Bibliography

References I

- Albert Atserias and Moritz Müller, *Automating resolution is np-hard*, 2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS), 2019, pp. 498–509.
- Hans L. Bodlaender and Dimitrios M. Thilikos, *Constructive linear time algorithms for branchwidth*, Automata, Languages and Programming (Berlin, Heidelberg) (Pierpaolo Degano, Roberto Gorrieri, and Alberto Marchetti-Spaccamela, eds.), Springer Berlin Heidelberg, 1997, pp. 627–637.
- Martin Davis, George Logemann, and Donald Loveland, *A machine program for theorem-proving*, Commun. ACM **5** (1962), no. 7, 394–397.
- Martin Davis and Hilary Putnam, *A computing procedure for quantification theory*, J. ACM **7** (1960), no. 3, 201–215.
- Armin Haken, *The intractability of resolution*, Theoretical Computer Science **39** (1985), 297–308, Third Conference on Foundations of Software Technology and Theoretical Computer Science.
- Petr Hliněný, Sang-il Oum, Detlef Seese, and Georg Gottlob, *Width Parameters Beyond Tree-width and their Applications*, The Computer Journal **51** (2007), no. 3, 326–362.
- Kensuke Imanishi, *An upper bound for resolution size: Characterization of tractable sat instances*, WALCOM: Algorithms and Computation (Cham) (Sheung-Hung Poon, Md. Saidur Rahman, and Hsu-Chun Yen, eds.), Springer International Publishing, 2017, pp. 359–369.

References II

Thanks for listening.