Exact Exponential Algorithms
- Under Dr. N.R. Aravind

Taha Adeel Mohammed

Indian Institute of Technology Hyderabad

May 1, 2024

Problem Statement

- Understand and analyse the different existing exponential algorithmic techniques.

Problem Statement

- Understand and analyse the different existing exponential algorithmic techniques.
- Study the PPSZ algorithm for solving the κ-SAT problem.

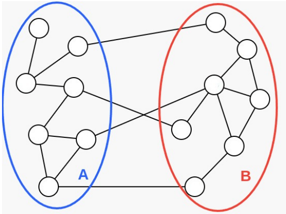
Problem Statement

- Understand and analyse the different existing exponential algorithmic techniques.
- Study the PPSZ algorithm for solving the $\textsc{k-SAT}$ problem.
- Attempt to improve the current best known time complexity upper bound for the $\textsc{Matching Cut}$ problem.

Matching Cut

MATCHING CUT Problem

- Given a graph $G = (V, E)$.
- Decide whether it can be partitioned into two sets A and B.
- $\forall\ v \in A$, v has ≤ 1 neighbour in B and vice versa.



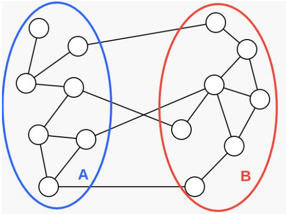
[1]Komusiewicz et al (2019). *Matching Cut: Kernelization, Single-Exponential Time FPT, and Exact Exponential Algorithms*. 10.1016/j.dam.2019.12.010.

[2]NR Aravind et al (2021). *An FPT algorithm for Matching Cut and d-cut*. arXiv:2101.06998

Matching Cut

Matching Cut Problem

- Given a graph $G = (V, E)$.
- Decide whether it can be partitioned into two sets A and B.
- $\forall\ v \in A$, v has ≤ 1 neighbour in B and vice versa.



Approaches

- SAT-free Branching Algorithm:[1] $O^*(1.3803^n)$
- Reduction to 3-SAT:[1] $O^*(1.3071^n)$
- FPT algorithm for d-CUT:[2] $O^*(2^{O(k \log k)})$ $\quad$ ($k =$ max edge-cut size)

[1] Komusiewicz et al (2019). *Matching Cut: Kernelization, Single-Exponential Time FPT, and Exact Exponential Algorithms*. 10.1016/j.dam.2019.12.010.

[2] NR Aravind et al (2021). *An FPT algorithm for Matching Cut and d-cut*. arXiv:2101.06998

k-SAT

k-SAT Problem

Given a CNF formula with max k literals per clause, decide if there exists an assignment that satisfies all clauses.

Matching Cut to 3-SAT

- We add the constraint that for each $v \in V$, and its two neighbours u_1, u_2, they cannot be in different parts of the matching cut.

$$\phi(G) = \bigwedge_{v \in V} \bigwedge_{u_1, u_2 \in N(v)} (v \vee \neg u_1 \vee \neg u_2) \wedge (\neg v \vee u_1 \vee u_2) \tag{1}$$

- $|Variables| = O(n)$, $|Clauses| = O(n^3)$

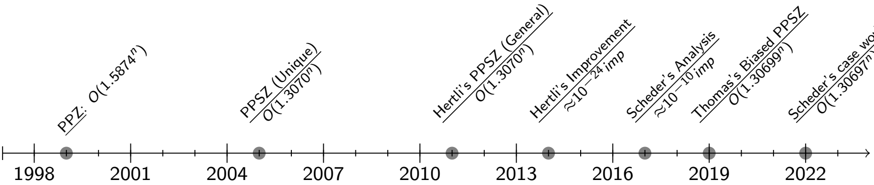
PPSZ Algorithm[3]

Def$^{\mathbf{n}}$: Let F be a CNF formula and $D \in \mathbb{N}$. We say F **D-implies** u if $\exists\ G \subseteq F$ with $|G| \leq D$ that implies u.

Algorithm

- Randomly choose permutation π of $vbl(F)$ and assignment $\beta \in \{0,1\}^n$
- Process variables in π order, setting each variable according to β, unless it is D-implied.

Timeline



[3] Ramamohan **P**aturi, **P**avel Pudlak, Michael E. **S**aks, and Francis **Z**ane. An improved exponential-time algorithm for k-SAT. J.ACM, 52(3):337364, 2005

PPSZ Analysis for Unique k-SAT[4]

<u>Def$^{\mathbf{n}}$</u>: A variable v is called **forced** if it is D-implied in F by α (the unique satisfying assgn.) and variables before v in π, else it is **guessed**.

$$\Pr_{\beta,\pi}[\text{ppsz returns } \alpha] = \mathbb{E}_\pi[2^{-|\text{Guessed}(F,\alpha,\pi,D)|}] \geq 2^{-\sum_x \Pr[x \text{ is guessed}]} \tag{2}$$

[4] Paturi, Pudlak, Saks, and Zane. *Chapter 6*: The PPSZ Algorithm*

PPSZ Analysis for Unique k-SAT[4]

Defn: A variable v is called **forced** if it is D-implied in F by α (the unique satisfying assgn.) and variables before v in π, else it is **guessed**.

$$\Pr_{\beta,\pi}[\text{ppsz returns } \alpha] = \mathbb{E}_\pi[2^{-|\text{Guessed}(F,\alpha,\pi,D)|}] \geq 2^{-\sum_x \Pr[x \text{ is guessed}]} \tag{2}$$

Critical Clause Tree

- T_x - Clause tree with var-label(root) $= x$. Each node u also has a clause-label(u) and assignment $\beta(u)$ based on the tree construction.
- Represents how x can become D-implied in F.
- Reachable(T_x, α, π): Set of all nodes u s.t. $\pi(v) \geq \alpha \ \forall \ v \in \text{path}(x, u)$.
- **Clause Tree Lemma:** $|\text{Reachable}(T_x, \pi(x), \pi)| \leq D \Rightarrow x$ is forced.

[4] Paturi, Pudlak, Saks, and Zane. *Chapter 6*: The PPSZ Algorithm*

PPSZ Analysis for Unique k-SAT[4]

Def$^{\mathbf{n}}$: A variable v is called **forced** if it is D-implied in F by α (the unique satisfying assgn.) and variables before v in π, else it is **guessed**.

$$\Pr_{\beta,\pi}[\text{ppsz returns } \alpha] = \mathbb{E}_\pi[2^{-|\text{Guessed}(F,\alpha,\pi,D)|}] \geq 2^{-\sum_x \Pr[x \text{ is guessed}]} \tag{2}$$

Critical Clause Tree

- T_x - Clause tree with var-label(root) $= x$. Each node u also has a clause-label(u) and assignment $\beta(u)$ based on the tree construction.
- Represents how x can become D-implied in F.
- Reachable(T_x, α, π): Set of all nodes u s.t. $\pi(v) \geq \alpha \ \forall \ v \in \text{path}(x, u)$.
- **Clause Tree Lemma:** $|\text{Reachable}(T_x, \pi(x), \pi)| \leq D \Rightarrow x$ is forced.

Using above clause tree, distinct label lemma, extending T'_x to T_∞, and expectation calculations, we get:

$$O^*(\text{ppsz}) = O^*(2^{(2\ln 2 - 1)n}) = O^*(1.30704^n) \tag{3}$$

[4]Paturi, Pudlak, Saks, and Zane. *Chapter 6*: The PPSZ Algorithm*

Hertli's Analysis for general k-SAT[5]

Definitions

- x is called a **frozen** variable if it has the same value in all satisfying assignments; **liquid** otherwise.
- $SL(F)$: Set of all satisfying literals. $|SL(F)| = |\text{frozen}| + 2|\text{liquid}|$

[5] Timon Hertli. *3-SAT Faster and Simpler - Unique-SAT Bounds for PPSZ Hold in General.* IEEE, 2011.

Hertli's Analysis for general k-SAT[5]

Definitions

- x is called a **frozen** variable if it has the same value in all satisfying assignments; **liquid** otherwise.
- $SL(F)$: Set of all satisfying literals. $|SL(F)| = |\text{frozen}| + 2|\text{liquid}|$

Modified PPSZ Algorithm

Instead of strictly following π, after each step, all D-implied variables are immediately set.

[5] Timon Hertli. *3-SAT Faster and Simpler - Unique-SAT Bounds for PPSZ Hold in General.* IEEE, 2011.

Hertli's Analysis for general k-SAT[5]

Definitions

- x is called a **frozen** variable if it has the same value in all satisfying assignments; **liquid** otherwise.
- $SL(F)$: Set of all satisfying literals. $|SL(F)| = |\text{frozen}| + 2|\text{liquid}|$

Modified PPSZ Algorithm

Instead of strictly following π, after each step, all D-implied variables are immediately set.

Theorem-1

If x is a frozen variable, then $p_{\text{guessed}}(F, x, \alpha, D) \leq S_k + \epsilon_k(D)$, where $S_k = \int_0^1 \frac{t^{1/(k-1)}-t}{1-t}$, and $\epsilon_k(D) \to 0$ as $D \to \infty$. $(S_3 = 2\ln 2 - 1)$

[5] Timon Hertli. *3-SAT Faster and Simpler - Unique-SAT Bounds for PPSZ Hold in General.* IEEE, 2011.

Hertli's Analysis for general k-SAT

AssignSL(F)

- Random process that generates a satisfying assignment.
- $\alpha = \{\}$; while $|\alpha| < n$: pick $\ell \in_R SL(F)$ and add it to α; return α;
- **Def^n**: $p(F, \alpha) =$ Probability that AssignSL(F) returns α.

$$p(F, \alpha) = \frac{1}{|SL(F)|} \sum_{\ell \in \alpha} p(F^{[\ell]}, \alpha) \tag{4}$$

Hertli's Analysis for general k-SAT

AssignSL(F)

- Random process that generates a satisfying assignment.
- $\alpha = \{\}$; while $|\alpha| < n$: pick $\ell \in_R SL(F)$ and add it to α; return α;
- **<u>Def$^{\mathbf{n}}$</u>**: $p(F, \alpha) =$ Probability that AssignSL(F) returns α.

$$p(F, \alpha) = \frac{1}{|SL(F)|} \sum_{\ell \in \alpha} p(F^{[\ell]}, \alpha) \tag{4}$$

Cost Function

- For variable x in F, cost is defined as

$$c(F, x) = \begin{cases} S_k & \text{if } x \text{ is liquid} \\ \sum\limits_{\alpha \in sat(F)} p(F, \alpha) p_{\text{guessed}}(F, x, \alpha, D) & \text{if } x \text{ is frozen} \end{cases} \tag{5}$$

- Cost of F is defined as $c(F) = \sum_{x \in vbl(F)} c(F, x)$.

$$\implies c(F) \leq nS_k \tag{6}$$

Hertli's Analysis for general k-SAT

Expectation of cost function (Theorem-2)

$$\mathbb{E}_\ell[c(F^{[\ell]})] \leq c(F) - |\text{liquid}|\frac{2S_k}{|SL(F)|} - |\text{frozen}|\frac{S_k}{|SL(F)|} \tag{7}$$

Success Probability

$$p_{\text{success}}(F) \geq 2^{-c(F)} \tag{8}$$

Proof by induction: Base case trivial.

$$p_{\text{success}}(F) = \frac{1}{2n}\sum_{\ell \in SL(F)} p_{\text{success}}(F^{[\ell]}) \;\geq\; \frac{1}{2n}\sum_{\ell \in SL(F)} 2^{-c(F^{[\ell]})}$$

$$\geq \frac{|SL(F)|}{2n}\mathbb{E}_\ell[2^{-c(F^{[\ell]})}] \;\geq\; \frac{|SL(F)|}{2n} 2^{-\mathbb{E}_\ell[c(F^{[\ell]})]}$$

$$\implies p_{\text{success}}(F) \geq 2^{\log\frac{|SL(F)|}{2n} - \mathbb{E}_\ell[c(F^{[\ell]})]} \;\geq\; 2^{-c(F)}$$

□

Matching Cut using P_3 Double Dominating Set

P_3 Double Dominating Set

- Let $S \subseteq V$ s.t. every P_3 of $G[V \setminus S]$ has a vertex with atleast 2 neighbours in S.
- **Proposition:** Let G be a n-vertex graph without leaves or adjacent degree-2 vertices. Then $\exists\ S$ s.t. $|S| \leq \lceil \frac{2n}{5} \rceil$, which can be found quickly.

Matching Cut using P_3 Double Dominating Set

P_3 Double Dominating Set

- Let $S \subseteq V$ s.t. every P_3 of $G[V \setminus S]$ has a vertex with atleast 2 neighbours in S.
- **Proposition:** Let G be a n-vertex graph without leaves or adjacent degree-2 vertices. Then $\exists\ S$ s.t. $|S| \leq \lceil \frac{2n}{5} \rceil$, which can be found quickly.

Algorithm

Brute-force vertices in S in $O(2^{|S|})$ and check for Matching Cut by reduction to 2-SAT:

1. For $v \in S$, add clause (x_v) or $(\neg x_v)$ based on $v \in A$ or $v \in B$.
2. For $v \in V \setminus S$ and $u_1, u_2 \in N(v)$,
   - If $x_{u_1} = x_{u_2}$, then $x_v = x_{u_1}$, i.e. add (x_v) or $(\neg x_v)$ appropriately.
   - Else if $x_{u_1} = x_{u_2}$, then $\forall\ u \in N(v) \setminus \{u_1, u_2\}$, we have the condition $x_u = x_v$, i.e. add $(x_v \vee \neg x_u) \wedge (\neg x_v \vee x_u)$.

Thank You!