Lecture 5

Uniform generation
- Uniformly generating satisfying assignments to DNF formula

Counting problems
- \#P

Approximate counting
- connection to uniform generation
Uniform sampling of satisfying assignments to DNF formula

DNF Formula:
“or of ands”

e.g. \( \varphi(x_1 \ldots x_n) = x_1 \overline{x}_2 x_3 \lor x_2 \overline{x}_3 x_4 x_5 \lor x_8 \overline{x}_{10} x_{11} \lor \ldots \)

Notation: implicit \( \land \)'s (we don’t bother to write them)

Task: Find satisfying assignment to \( \varphi \)
easy!
pick one term \( \lor \) set literals in it to true (satisfied if \( \exists \) term st. not both \( x_i \lor \overline{x}_i \) in it)

Task: Find random satisfying assignment to \( \varphi \)
uniform over all sat assignments
Is it doable???

**Special case:**

**Only one conjunction**

\[ F = y_1 y_2 \ldots y_k \quad \text{for } y_i \in \{ x_i, \overline{x_i}, x_i \overline{x_i} \} \]

**e.g.** \( F = x_1 \overline{x_2} x_3 \)

sat assignments \(=\) any assignment st.

\[ X_1 = T, \ X_2 = F, \ X_3 = T \]

random satisfying assignment to \( F \):

Let \( X_1 = T, \ X_2 = F, \ X_3 = T \)

\( \ast \) pick \( X_4 \ldots X_n \) randomly \( \in \{ T, F \} \)

in general, satisfy literals in \( F \)

\( \ast \) pick other settings randomly
Two Conjunction Case:

Algorithm Attempt:

- Pick $i \in \{1,2,3\}$
- Set vars in conjunction $i$ to "true"
- Set other vars randomly

Example: $x_1 x_2 \lor x_3$

- Pick 1
- Set $x_1 = x_2 = T$
- Set $x_3 = T$

Not uniform:

1) 2nd conjunction has more sat assignments
2) Some assignments can be chosen multiple ways

Pr[output $TTF$] = $\frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$

Pr[output $TFT$] = $\frac{1}{2} \cdot \frac{1}{4} = \frac{1}{8}$

Pr[output $TTT$] = $\frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$

$Pr[TTF] = \frac{1}{4}$
$Pr[TFT] = \frac{1}{8}$
$Pr[TTT] = \frac{3}{8}$
main ideas to fix algorithm:

1) choose conjunction proportionally to \# sat assignments

2) if assignment can be output in >1 way, "correct" for it

"rejection sampling"

Let \( \hat{A}_i \subseteq \{ x = (x_1 \ldots x_n) \mid x \text{ satisfies } C_i \} \) assignments that satisfy clause \( i \)
Algorithm: Input: $\Phi = \bigvee_{i=1}^{m} C_i$

Let $A_i \leftarrow \exists \bar{x} = (x_1 \ldots x_n) \mid \bar{x} \text{ satisfies } C_i \exists$

Repeat
  Pick $i$ with prob $\frac{|A_i|}{\sum_{j \in \Phi} |A_j|}$
  Pick uniform assignment $\bar{b}$ in $A_i$
  Let $t_5 \leftarrow |\exists j \mid \bar{b} \text{ satisfies } A_j \exists|$
  Output $\bar{b}$ with prob $\frac{1}{t_5}$

Until succeed
Uniformity:

\[ \forall b \text{ st. } \overline{b} \text{ satisfies } \varnothing : \]

\[
\Pr[\text{output } b \text{ in round } i] = \frac{1}{t_b} \sum_{j \in [m], \text{ s.t. } b \in S_j} \Pr[\text{pick } j \text{ in round } i] \cdot \frac{1}{|S_j|}
\]

\[
= \frac{1}{t_b} \sum_{j \in S_j, \text{ s.t. } b \in S_j} \frac{|S_j|}{|S_j|} \cdot \frac{1}{|S_j|}
\]

\[
= \frac{1}{t_b} \frac{\sum_{k \in [m]} \frac{|S_k|}{|S_k|}}{k} = \frac{1}{m} \sum_{k \in [m]}|S_k|
\]

same for all \(b\) that satisfy \(\varnothing\)

Runtime:

\[
\Pr[\text{loop succeeds}] \geq \frac{1}{\max t_b} = \frac{1}{m}
\]

\[E[\# \text{ loops until succeeds}] \leq m\]

time per loop is \(\text{poly}(m+n)\)
Counting Problems

$\#P = \text{class of problems that count}$
$\# \text{ accept paths in poly-time non deterministic Turing machines}.$

$\#P$-complete:
- in $\#P$
- every problem in $\#P$ has Turing reduction $\leq_T$ to it
  poly-time reduction

$\#\text{SAT} : \#\text{ of assignments satisfying Boolean formula } \phi$
$\#P$-complete!
Is \( \#\text{DNF} \) easier?

Not if \( P \neq \text{NP} \)

Why?

Given \( \phi \) in \( \text{CNF} \)

\( \phi \) is sat iff \( \overline{\phi} \) has \( >1 \) unsat assignments

\( P=\text{NP} \iff \) ability to exactly count in poly time \( \iff \) ability to exactly count \( \#\text{DNF} \) in poly time

\( \#\text{DNF} \) is \( \#\text{P}\text{-complete} \)
Approximate Counting

Fully polynomial randomized approximation scheme (FPRAS)

\[
\text{Given } \mathcal{P}, \varepsilon \\
\text{s.t. } z = \# \text{ sat assignments to } \mathcal{P}
\]

\[
\text{Output } y \text{ s.t. } \\
\frac{z}{1+\varepsilon} \leq y \leq z(1+\varepsilon)
\]

with prob \( \geq 3/4 \)

Hope: runtime poly in \( |\mathcal{P}|, \frac{1}{\varepsilon} \)

pset 1 problem 1:

algorithm that satisfies "hope"

\[ \Rightarrow \text{ poly in } |\mathcal{P}|, \frac{1}{\varepsilon}, \log \frac{1}{\delta} \]

approx error

\[ \propto \text{ prob of too much approx error} \]

"Confidence"
FPRAS for SAT?

FPRAS for SAT $\Rightarrow$ ptime algorithm for SAT:

**Algorithm for SAT:** Given formula $\varphi$

- Call FPRAS on $\varphi$ with $\varepsilon = \frac{1}{2}$
- if output $> 0$ output "Satisfiable"
- else output "unsatisfiable"

**Correctness**

- if $\varphi$ satisfiable, $\#\varphi \geq 1$ so
  - $y > \frac{1}{1+\varepsilon} > 0 \Rightarrow$ output "sat"
- if $\varphi$ unsatisfiable, $\#\varphi = 0$ so
  - $y = 0 \Rightarrow$ output "unsat"
Exact vs. Approx Counting

Counting # SAT assignments to CNF is #P-complete

DNF

perfect matchings in graph

spanning trees in graph is in polytime

Is it hard to approx count?

CNF hard

DNF polytime ← today

Matching polytime

Spanning trees polytime

Your favorite problem?
Fully polynomial randomized approximation scheme (FPRAS)

Given \( \emptyset, \epsilon \)

\[ \text{s.t. } z = \# \text{ sat assignments to } \emptyset \]

Output \( y \) s.t.

\[ \frac{z}{1+\epsilon} \leq y \leq z \cdot (1+\epsilon) \]

with \( \text{prob} \geq 3/4 \)

Approx counting for DNF: 

Will use:

1. Uniform generation of DNF sat assignments
2. "Downward self-reducibility" of DNF

Downward self-reducibility: (dsr)

Can compute problem by solving smaller subproblems & putting together answers via poly time computation.
Why is $\# -\text{DNF}$ dsr.?

\[ \# \phi(x_1 \ldots x_n) = \pm \phi(x_1 = T, x_2, \ldots x_n) + \]

both are still DNFs

but in $n-1$ vars.

\[ \# \phi(x_1 = F, x_2, \ldots x_n) \]

\[ \begin{aligned}
\text{e.g.,} & \quad \#(x_1 \bar{x}_2 \lor x_1 x_2 \lor \bar{x}_2) \\
& = \#(\bar{x}_2 \lor x_2 \lor \bar{x}_2) \\
& \quad + \#(\bar{x}_2) \\
\end{aligned} \]

$\leq \#\text{settings}$

where $x_1 = T$

$\leq \#\text{settings}$

where $x_1 = F$
Downward Self-Reducibility Tree

\[ F = \neg \varphi(x_1 \ldots x_n) = F_0 + F_1 \]

\[ F_0 = \neg \varphi(F_0, F_1, x_3 \ldots x_n) = F_{00} + F_{01} \]

\[ F_1 = \neg \varphi(T_1, x_2 \ldots x_n) = F_{10} + F_{11} \]

Each node is sum of children.

Leaves either 1 = true, 0 = false.

DNF in 0 vars \( \Rightarrow \) either True or False.
Example

\[ \# (x_1 \overline{x_2} \lor x_1 x_2 \lor \overline{x_2}) = 3 \]

\[ \# (\overline{x_2} \lor x_2 \lor \overline{x_2}) = 2 \]

\[ \# (\overline{x_2}) = 1 \]
Approximate Counting Algorithm for #DNF

Let $S_i = \frac{F_i}{F} \Rightarrow F = \frac{F_i}{S_i}$

$\left\lfloor \text{Fraction of sat assignments st. } x_i = T \right\rfloor$

Main insight: for DNF, we can estimate $S_i$ via sampling!

- Uniformly generate $K$ sat assignments
- $S_i \left\lfloor \text{ with } x_i = T \right\rfloor \left\lceil \text{ for } DNF\right\rfloor$!

But how do we compute $F_i$?

Recursively!

$F_i = \frac{F_{ii}}{S_{ii}}$ \leftarrow \text{ estimate}$
So \( F = \frac{F_{b_1}}{S_{b_1}} = \frac{F_{b_1}b_2}{S_{b_1}S_{b_1}b_2} = \frac{F_{b_1}b_2b_3}{S_{b_1}S_{b_2}S_{b_3}b_2b_3} \)

\[
= \frac{1}{\prod_{i=1}^{n} S_{b_i} \cdot b_i}
\]

**Potential Difficulties:**

1. If \( F_{b_1...b_n} = 0 \) this doesn't work
2. Is approximation of \( S_{b_1...b_i} \)'s good enough? Only get additive estimates

**Idea:** Always take path of "larger" child

Claim if always pick \( b_i \) s.t. \( F_{b_1...b_i} > F_{b_1...\bar{b}_i} \) then always reach SAT assignment leaf.

(\( S_{b_1...b_n} = 1 \))

\[ \text{might guess wrong when both have lots of SAT assignments but soon will show that is ok} \]
estimate each $S_{b_1 \ldots b_i}$ to within $\frac{\varepsilon}{4n}$

additive error (using Chernoff bounds, need only $\text{poly}(\frac{\varepsilon^2}{\varepsilon}, \log \frac{1}{\varepsilon})$ samples to get error $< \frac{1}{4n}$)

$$1 + r + \frac{\varepsilon}{4n} \leq r\left(1 + \frac{\varepsilon}{4n}n\right) \leq r\left(1 + \frac{\varepsilon}{2n}\right)$$

union bound over all $i$ to get prob of error $< \frac{1}{4}$

* Issue: might be estimating $1-r$ if pick wrong path. We will ignore this for now.

$$r - \frac{\varepsilon}{4n} \geq r\left(1 - \frac{\varepsilon}{4nr}\right) \geq r\left(1 - \frac{\varepsilon}{4n}\right)$$

Claim:

$$\text{output} \leq \frac{F_{b_1}}{S_{b_1}} \leq \frac{F_{b, b_2}}{S_{b_1} S_{b, b_2}} \leq \ldots \leq \frac{1}{\prod S_{b_1 \ldots b_n}}$$

$$\leq \left(1 + \frac{\varepsilon}{4n}\right)^n = F \cdot \left(1 + \frac{\varepsilon}{4n}\right)^n \leq F\left(1 - \frac{\varepsilon}{2}\right)^n \leq 1 + \frac{\varepsilon}{2}$$

Similarly, $\text{output} \geq \frac{F}{1 + \varepsilon}$.
Recursive Algorithm

- estimate $S_0, S_1$ from uniform generated SAT assignments
- let $b_i \leftarrow \text{argmax } \exists S_0, S_1$
- recurse on $F_{b_i}$

$\Pr[\text{algorithm fails}] \leq \sum_{i=1}^{n} \Pr[\text{estimate bad}] \leq n \cdot \frac{1}{8n} \leq \frac{1}{8}$. 
Works for any d.s.r. problem!

poly time (almost)-uniform-generation of solutions

\[ \uparrow \]

poly time approximate counting of #sols

Theorem [Jerrum, Valiant, Vazirani] for any problem in \( \text{NP} \)

that is d.s.r.:

ptime approx counting of #sols \iff ptime almost uniform generation
(easier case)

(Perfect) counting for $\# \text{DNF} \Rightarrow$

(perfect) Uniform generation

Recursive algorithm:

at $b_1 \ldots b_n$,

use (perfect) counter to compute

$F_0 = \overline{F_{b_1 \ldots b_n} 0}$

$F_1 = F_{b_1 \ldots b_n} 1$

go left with prob $\frac{r_0}{r_0 + r_1}$

and right o.w.

Claim (1) always reach SAT assignment

since never take branch with 0 SAT assignments underneath

(2) $\Pr[\text{output } b_1 \ldots b_n] = \frac{F_{b_1}}{F} \cdot \frac{F_{b_1 b_2}}{F_{b_1}} \cdot \frac{F_{b_1 b_2 b_3}}{F_{b_1 b_2}} \cdot \ldots \cdot \frac{1}{F_{b_1 \ldots b_n}}$

$= \frac{1}{F} \leftrightarrow$ same for every SAT assignment
Question: what if only have approx counter?

Answer: \[
\text{RHS} \leq \frac{1}{F} \left( \frac{1+1}{3-1} \right)^n \leq \frac{1}{F} \cdot \frac{1}{3-1} \\
\text{if choose } \varepsilon > \frac{3}{2n} \\
\Rightarrow \text{close to uniform generation of sat assignments}
\]