Lecture 9

Derandomization via method of conditional expectations

Uniform generation

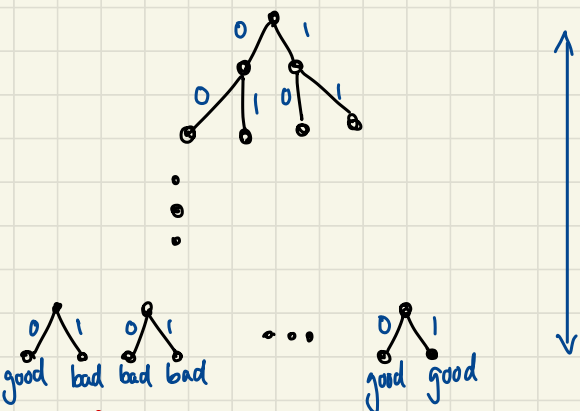
- Uniformly generating satisfying assignments to DNF formula

Counting problems

- #P

Derandomization via the method of conditional expectations

Idea: view coin tosses of algorithm as path down tree of depth m = # coin tosses



alternatively: $c_{00\ldots0}$ $c_{00\ldots1}$ $c_{00\ldots10}$ c_{00-11} $c_{11\ldots10}$ c_{11-1}
cut values

good = correct / reach witness / good approximation / pass ...

good randomized algorithm $\Rightarrow$ most leaves good

idea: find a good path to leaf "bit-by-bit"

more formally:

Fix randomized algorithm A

input x
m = # random bits used by A on x

for $1 \le i \le m$ & $r_1 \cdots r_i \in \{0,1\}$

let $p(r_1 \cdots r_i)$ = fraction of continuations that end in "good" leaf (good approx)

set 1st i bits to $r_1 \cdots r_i$
pick $(i+1)^{st}$ to m^{th} bits randomly

$e(r_1 \cdots r_i)$ = average cut value if set first i nodes to $r_1 \cdots r_i$

$$p(r_1 \cdots r_i) = \frac{1}{2} \cdot p(r_1 \cdots r_i 0) + \frac{1}{2} \cdot p(r_1 \cdots r_i 1)$$

by averaging, $\exists$ setting of r_{i+1} to 0 or 1 s.t. $p(r_1 \cdots r_{i+1}) \ge p(r_1 \cdots r_i)$

can we figure out which one?

if $p(r_1 \cdots r_{i+1}) \geq p(r_1 \cdots r_i) \quad \forall i$

then $p(r_1 \cdots r_m) \geq p(r_1 \cdots r_{m-1}) \geq \cdots \geq p(r_1) \geq$ fraction of good paths $\geq 2/3$

arbitrary const $\geq 1/2$

$p(r_1 \cdots r_m)$: this is a leaf so value is 1 or 0 but if $\geq 2/3$ it must be 1

$\Longrightarrow$ picking best branch leads to a good leaf

main issue:

how do we choose best r_{i+1} setting at each step?

Example Max cut (second way to derandomize)

(recall) algorithm:

flip n coins $r_1 \cdots r_n$

put node i in S if $r_i = 0$ & T if $r_i = 1$

Output S, T

(recall)

Analysis:

u, v on opposite sides of cut

Let $1_{u,v} = \begin{cases} 1 & \text{if } r_u \neq r_v \\ 0 & \text{o.w.} \end{cases}$

$$\text{Cut size} = \sum_{(u,v) \in E} 1_{u,v}$$

$$E[\text{cut size}] = E\left[\sum_{(u,v) \in E} 1_{u,v}\right]$$

$$= \sum_{(u,v) \in E} E[1_{u,v}] = \sum_{(u,v) \in E} \Pr[1_{u,v} = 1]$$

$$= \sum_{(u,v) \in E} \Pr\left[(r_u = 1 \text{ and } r_v = 0) \text{ or } (r_u = 0 \text{ and } r_v = 1)\right]$$

$$= |E| \cdot \left[\tfrac{1}{4} + \tfrac{1}{4}\right] = \frac{|E|}{2}$$

So expect $\frac{1}{2}$ the edges to cross cut!

Note: $E[\text{cut size}] = \frac{|E|}{2} \Rightarrow \exists$ cut of size $\frac{|E|}{2}$

$E[\text{cut size}]$: average cut size produced by algorithm

cut of size $\frac{|E|}{2}$: must be one that is at least as big as the average

derandomization:

$$e(r_1 \cdots r_i) = E_{R_{i+1} \cdots R_n}\left[\, |\text{cut}(S,T)| \text{ given } r_1 \cdots r_i \text{ choices made} \,\right]$$

e (no choices fixed yet) $\geq \frac{|E|}{2}$ (previous lecture)

how do we calculate $e(r_1 \cdots r_{i+1})$?

Let

$S_{i+1} = \{\text{nodes } j \mid j \leq i+1, r_j = 0\}$

$T_{i+1} = \{\text{nodes } j \mid j \leq i+1, r_j = 1\}$

S & T so far

$V_{i+1} = \{\text{nodes } j \mid j \geq i+2 \text{ \& } j \leq n\}$ undecided

so:

main insight #1

fact $e(r_1 \cdots r_{i+1}) = (\#\text{ edges between } S_{i+1} \text{ \& } T_{i+1}) + \frac{1}{2}(\#\text{ edges touching } V_{i+1})$

(follows from same reasoning as last lecture)

main insight #2

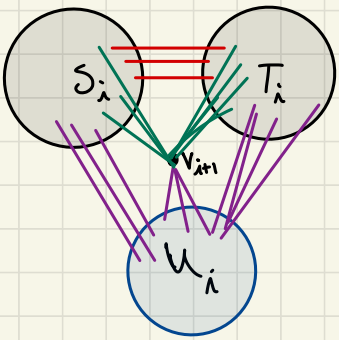
Note: don't need to calculate $e(r_1 \cdots r_{i+1})$

just need to figure out which is bigger

$e(r_1 \cdots r_i 0)$ or $e(r_1 \cdots r_i 1)$

how do we do this?

- # edges touching V_{i+1} same for both
- S_{i+1}, T_{i+1} & S_i, T_i differ only in placement of v_{i+1} & edges touching v_{i+1} going to $S_{i+1} T_{i+1}$ (edges to U_{i+1} are same no matter which choice made)



to maximize place node $i+1$ to maximize cut size:

Compare

#edges between V_{i+1} & S_i

VS. " " " " " T_i

$\Longrightarrow$ Can deterministically pick which choice gives bigger # edges

$\Longrightarrow$ if do this for each i, get solution which is $\geq$ expected value in deterministic way

yields:

Greedy Algorithm

1) $S \leftarrow \phi$, $T \leftarrow \phi$

2) For $i = 0 \ldots n-1$

place v_i in S if #edges between v_i & T $\geq$ " " " " " S

else place v_i in T

New topic: uniform generation

Uniform sampling of satisfying assignments to DNF formula

DNF Formula:

"or of ands"

e.g. $\varphi(x_1 \cdots x_n) = x_1\bar{x}_2x_3 \vee x_2\bar{x}_3x_4x_{10} \vee x_8\bar{x}_{10}x_{11} \vee \ldots$

Notation: implicit $\wedge$'s (we don't bother to write them)

Task: Find satisfying assignment to φ

easy!

pick one term & set literals in it to true

(satisfied if $\exists$ term s.t. not both x_i & $\bar{x}_i$ in it)

Task: Find random satisfying assignment to φ

↑ uniform over all sat assignments

Is it doable???

Easy special case: Only one conjunction

$$F = y_1 \wedge y_2 \wedge \dots \wedge y_k \quad \text{for } y_i \in \{x_1, \bar{x}_1, x_2, \bar{x}_2, \dots\}$$

e.g. $F = x_1 \bar{x}_2 x_3$

sat assignments $\equiv$ any assignment s.t. $x_1 = T$, $x_2 = F$, $x_3 = T$

random satisfying assignment to F:

Let $x_1 = T$, $x_2 = F$, $x_3 = T$

& pick $x_4 \dots x_n$ randomly $\in \{T, F\}$

in general, satisfy literals in F

& pick other settings randomly

Two Conjunction Case:

Algorithm Attempt:

pick $i \in \{1,2\}$

set vars in conjunction i to "true"

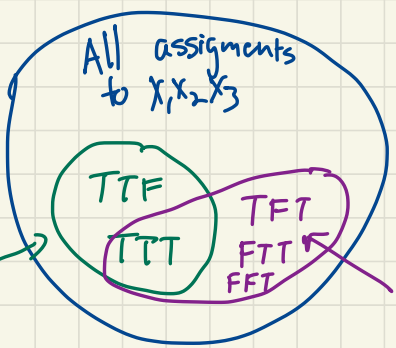
set other vars randomly

example: $x_1 x_2 \vee x_3$

pick 1

set $x_1 = x_2 = T$

set $x_3 = T$



satisfy x_1, x_2

satisfy x_3

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

(Prob pick 1; Prob pick $x_3 = T$)

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

(prob pick 2; prob pick $x_1 = T$, $x_2 = F$)

$Pr[\text{output } TTT] =$

$\frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{4} = 3/8$

pick 1; pick $x_3 = T$; pick 2; pick $x_1 = T$, $x_2 = T$

Two problems which make it not uniform:

1) 2nd conjunction has more sat assignments

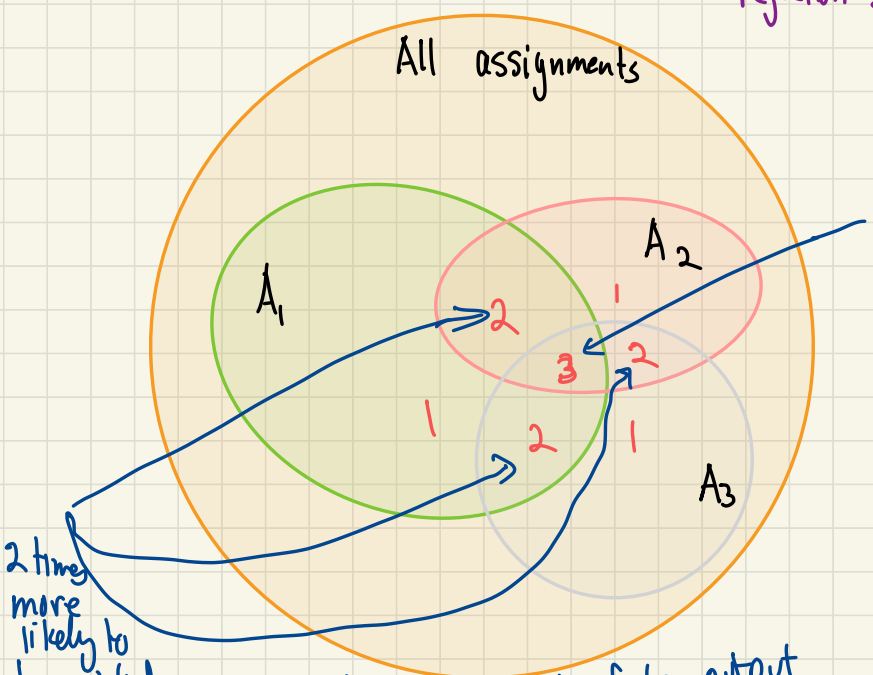
2) some assignments can be chosen multiple ways

main ideas to fix algorithm:

1) choose conjunction proportionally to # sat assignments

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

"rejecton sampling"



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

assignments that satisfy clause i

Algorithm: Input: $\phi = \bigvee_{i=1}^{m} C_i$ ← conjunctions

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

Repeat

Pick i with prob $\frac{|A_i|}{\sum_{j=1}^{m} |A_j|}$

we already saw → how to do this

Pick uniform assignment $\bar{b}$ in A_i

Let $t_{\bar{b}} \leftarrow |\{ j \mid \bar{b} \text{ satisfies } A_j \}|$ ← ≥ 1 since $\bar{b}$ satisfies A_i

Output $\bar{b}$ with prob $\frac{1}{t_{\bar{b}}}$

Until succeed

Uniformity:

$\forall\ \tau$ s.t. τ satisfies ϕ:

$$\Pr[\text{output } \tau \text{ in round } i] = \frac{1}{t_\tau} \sum_{\substack{k \in [m] \\ \text{s.t } \tau \in A_k}} \Pr[\text{pick } k \text{ in round } i] \cdot \frac{1}{|A_k|}$$

$$= \frac{1}{t_\tau} \sum_{\substack{k \text{ s.t.} \\ \tau \in A_k}} \frac{|A_k|}{\sum_{j=1}^{m} |A_j|} \cdot \frac{1}{|A_k|}$$

$$= \frac{1}{t_\tau} \frac{t_\tau}{\sum_j |A_j|} = \frac{1}{\sum_j |A_j|}$$

same for all τ that satisfy ϕ

Runtime:

$$\Pr[\text{loop succeeds}] \geq \frac{1}{\max t_\tau} \geq \frac{1}{m}$$

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

time per loop is poly $(m+n)$

Counting Problems

#P = class of problems that count # accept paths in poly-time non deterministic Turing machines.

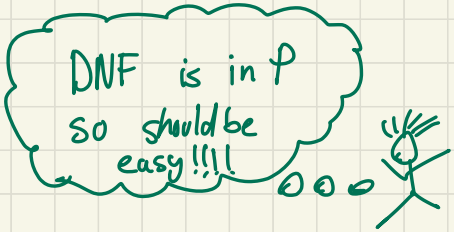
#P-complete:

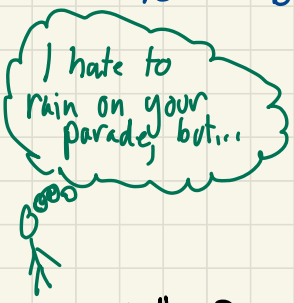
- in #P
- every problem in #P has Turing reduction / poly-time reduction to it

#SAT: # of assignments satisfying Boolean formula φ

#P-complete!

Is #DNF easier?





Not if $P \neq NP$

DeMorgan's law:

$\overline{(A \vee B)} = \bar{A} \wedge \bar{B}$

clause → conjunction

Why?

Given ϕ in CNF

ϕ is sat iff $\overline{\phi}$ has >1 unsat assignments

ϕ: ↑ CNF

$\overline{\phi}$: ↑ DNF

$P = NP \Leftarrow$ ability to exactly count CNF in poly time $\Leftarrow$ ability to exactly count DNF in poly time

#DNF is #P-complete

Approximate Counting

Fully polynomial randomized approximation scheme (FPRAS)

Given ϕ, ε

s.t. $z = \#$ sat assignments to ϕ

Output y s.t.

$$\frac{z}{1+\varepsilon} \leq y \leq z \cdot (1+\varepsilon)$$

with prob $\geq 3/4$

Hope: runtime poly in $|\phi|, \frac{1}{\varepsilon}$