CPSC 536F Jan 20

Last time:

How many formulas size s

on n Boolean variables ...

Estimate:

$\#\text{formulas}(s,2)$

$$\leq (\#\text{trees}) \cdot \left(\begin{array}{l}\text{setting of } s \text{ leaves}\\ \text{to}\\ x_1,\ldots,x_n,\neg x_1,\ldots,\neg x_n\end{array}\right) \left(\begin{array}{l}\text{setting of interior vertex to}\\ \wedge / \vee\end{array}\right)$$

we got

$$s^{s-1}(2n)^{s}(2)^{s-1}$$

We compare to $2^{(2^n)}$

Roughly: s size around 2^n

(range $2^n/n$, $2^n/n^3$, ...)

$$\boxed{s^{s-1}} \quad \boxed{(2n)^s} \quad \boxed{(2)^{s-1}} \quad \text{vs} \quad 2^{(2^n)}$$

$$f_1(n,s) \quad f_2(n,s) \quad f_3(n,s)$$

goal: set s s.t.

$$\log_2(f_1 \, f_2 \, f_3) \qquad\qquad \log_2\left(2^{(2^n)}\right)$$

$$\text{vs} \qquad \|$$

$$\log_2(\qquad\qquad) = o(2^n) \;???$$

So

$$\log_2 (f_1 f_2 f_3)$$

$$= \log_2 (f_1) + \log_2 (f_2) + \log_2 (f_3)$$

say we want

(1) set S around 2^n

(2) get

$$\log_2 f_1 + \log_2 f_2 + \log_2 f_3$$

$$\leq C \left(\overbrace{\text{something}}^{f(n)} \text{ simple} \right)$$

Then we want $f(n) = o(2^n)$

or

$$\lim_{n \to \infty} \frac{f(n)}{2^n} = 0$$

$g_1 = \log_2 f_1 = s-1 \quad \log s$

$g_2 = \log_2 f_2 = s \quad \log(2n)$

$g_3 = \log_2 f_3 = s-1 \quad \log 2$

s roughly 2^n

$\log_2 s$ roughly n

$g_2, g_3 = o(g_1)$

$g_1 + g_2 + g_3$

$$= g_1 (1 + o(n))$$

$$g_1 \le g_1 + g_2 + g_3 \le g_1 (1 + o(1))$$

So $$\lim_{n\to\infty} \frac{g_1 + g_2 + g_3}{g_1} = 1$$

best possible in comparing g_1, g_2, g_3

Focus on $g_1 = (s-1)\log s$

Simplify drop -1 from g_1

$$g_1 \leq s \log s$$

$$\lim_{n\to\infty} \frac{(s-1)\log s}{s \log s} = 1$$

$$g_1 + g_2 + g_3 = (s \log s)(1 + o(n))$$

$=$

So you want to ~~s~~ choose $s = s(n)$
s.t.

$$(s \log s)(1+o(n)) \leq o(2^n)$$

Then roughly

$$\boxed{s \log s \text{ to be } 2^n}$$

So

$$s \log s = 2^n$$

set equal to see the limit of what we can achieve

If

$$\log_2 s + \log_2 \log_2 s = n$$

$$\boxed{\log_2 s = n - \log_2 \log_2 s}$$

s near 2^n

$\log \log(s)$ very near $\log \log n$

$$\log_2 s = n - \log_2 \log_2 (s)$$

$$\log_2 (\log_2 s)$$

roughly n $\longrightarrow$ $\left(n - \log_2 \underbrace{\log_2 s}_{\text{roughly } n} \right)$ roughly $\log_2 n$

$\log_2 s$ "roughly" $n - \log_2 n$

Take

$$\log_2 s^* = n - \log_2 n$$

$$S^* = 2^{(n - \log_2 n)} = 2^n/n$$

We learn (by experience)

$$S^*(n) = 2^n/n$$

then

$$S^*(n) \log S^*(n)$$

$$= (2^n/n)(\log_2(2^n/n))$$

$$(2^n/n)(n - \log_2 n)$$

$$= \left(\frac{2^n}{n}\right)(n)\left(1 - \frac{\log_2 n}{n}\right)$$

$$= 2^n(1 + o(1))$$

"Trick" or "Method"

If $S^*(n) = 2^n/n$

then

$$S^*(n) \log S^*(n)$$

$$= 2^n (1 + o(1))$$

$$\left(\begin{array}{l} S^*(n) = 2^n/n \\ \\ n \; S^*(n) = 2^n \end{array} \right)$$

Similarly if

$$S^*(n) \log S^*(n) = g$$

g any function of n ~~grows~~ ∞ as $n \to \infty$

Version of this trick/method!

$$S^*(n) = \left(\frac{2^n}{n}\right) c, \quad c < 1$$

then

log

$$S^*(n) \log(S^*(n))$$

$$\sim \left(\frac{2^n}{n} c\right)(n - \lg n + \lg c)$$

$$\sim 2^n c \, (1 + o(1))$$

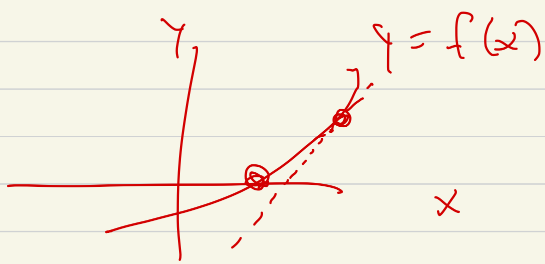
More formally: One of a set of tools

$$S^*(n) = \frac{2^n}{n} C$$

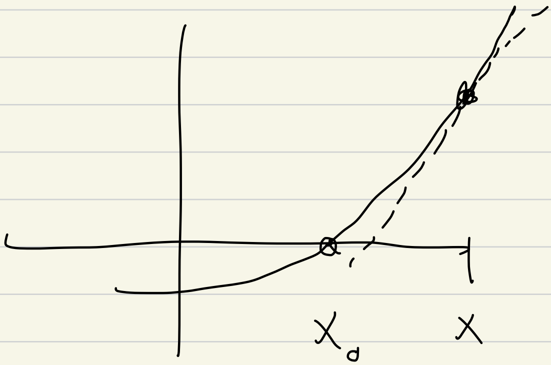
then

$$\lim_{n \to \infty} \frac{S^*(n) \log S^*(n)}{2^n} = C$$

Similar! Newtons method:



For x near root x_0



$\Phi(x)$

Then

$$x_0 \simeq x - \frac{f(x)}{f'(x)}$$

certainly

$$x_0 - \frac{f(x_0)}{f'(x_0)} = x_0 - \frac{0}{\cdot}$$

$$= x_0$$

$\Phi(x_0) = x_0$, hope

x near x_0

$\Phi(x)$ closer to x_0

$\Phi(\Phi(x))$ even closer ...

as long as $|\Phi'(x_0)| < 1$

then if Φ is differentiable and

then $\Phi \cdots \Phi(x) \to x_0$. x near x_0

Back to complexity

$$\left(\begin{array}{c}\#\text{ formulas}\\ \text{size } S\\ \text{on } n\\ \text{variables}\end{array}\right) \leq \left(\begin{array}{c}\#\text{ binary trees}\\ \text{size}\\ S\end{array}\right)(2n)^S(2)^{S-1}$$

C^S

Claim:

$$\begin{pmatrix} \text{\# binary} \\ \text{trees} \\ \text{size } s \end{pmatrix} \leq C^s$$

some constant C

=

If so:

$S(n)$ roughly 2^n

take

$$\log_2 C^s$$

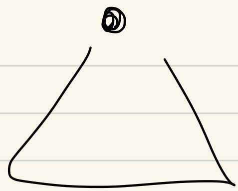
+

$$\longrightarrow \log_2 (2n)^s = \log_2(2^s) + \log_2(n^s)$$

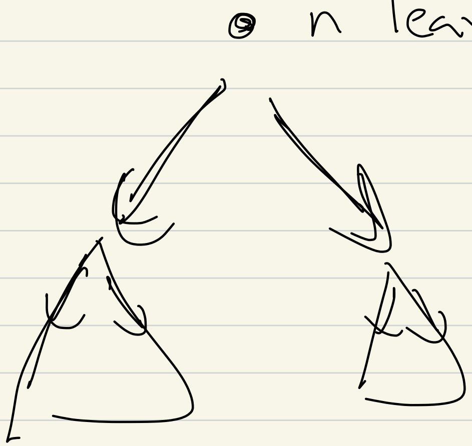
becomes dominant

+

$$\log_2 (2^s)$$



binary tree on n leaves

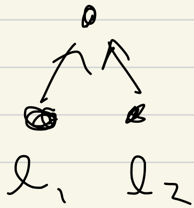


n leaves at root

$1 \le k \le n-1$

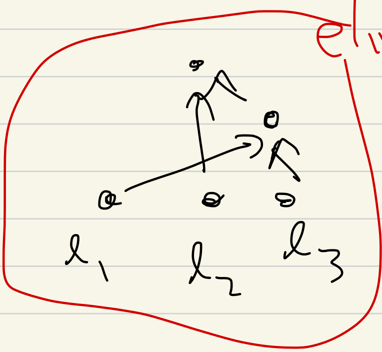
size k size $n-k$

$B(n)$ = # binary trees on n leaves

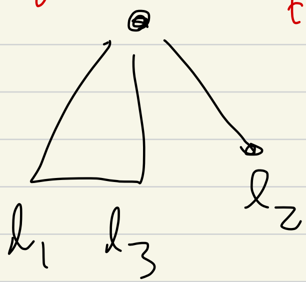


$B(2) = 1$

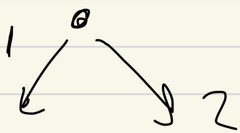
eliminate by going downwards from root

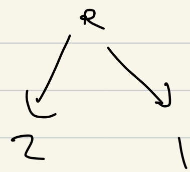


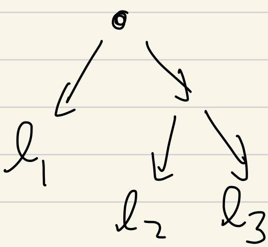
but

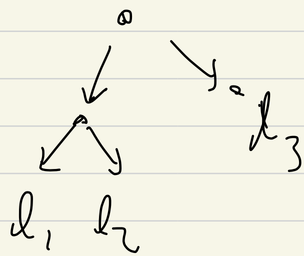


$B(3)$

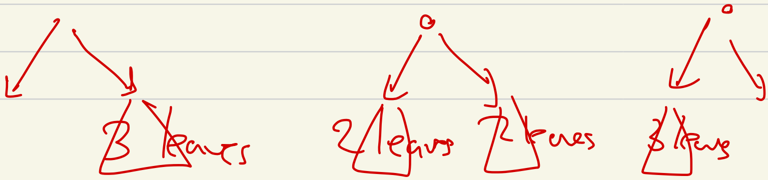








$B(4)$:



Could say

$$B(n) \leq B(n-1) + B(2)B(n-2) + \cdots + B(n-2)B(2) + B(n-1)$$

or better

$$B(n) \leq B(n-1) + B(2)B(n-2) + \cdots + B\left(\left\lfloor \frac{n}{2} \right\rfloor\right) B\left(\left\lceil \frac{n}{2} \right\rceil\right)$$

$$B(7) \leq B(6) + B(2)B(5) + B(3)B(4)$$

stop

$$B(8) = B(7) + B(2)B(6)$$

$$+ \cdots B(3)B(5)$$

$$+ B(4)B(4)$$

then stop

---

this recurrence

$$C(1) = C(2) = 1$$

$$C(n) = C(n-1) + C(2)\,C(n-2)$$

$$+ \cdots + C(n-2)\,C(2)$$

$$+ C(n-1)\,C(1)$$

"Catalan Numbers"

Regardless,

Claim: $B(n) \leq C^n$

$=$

$\log_2(\text{Formula Size}(n,s))$

$\leq \log_2(n^s)(1+o(1))$

Claim: If $s \leq \left(2^n/\log_2 n\right)c$

$c < 1$

$\log_2(n^s) = s \log_2 n$

$\sim 2^n \cdot c$

Solving

$t^*(n)$ roughly :

$$t^*(n) \log_2 n \sim 2^n$$

$$t^*(n) = 2^n / \log_2 n$$

and

$$t^*(n) = \frac{2^n}{\log_2 n} \cdot c \quad (c < 1)$$

$$t^*(n) \log_2 n \sim 2^n \cdot c$$

5 min break

10:23 – 10:33

We had:

$$\log_2 F(n,s) \approx 2^n c$$

thm with s roughly 2^n

last class

$$F(n,s) = s^s (\text{const})^s$$

we take $s \sim \frac{2^n}{n} c$

This class

$$\log_2 \hat{F}(n,s) \sim 2^n \cdot c$$

with s roughly 2^n

$$\tilde{F}(n,s) = n^s (\text{const})^s$$

$$s \sim \frac{2^n}{\log n} c$$

Homework!

$G(n,s) = \#\left\{\begin{array}{l}\text{straight-line programs}\\ \text{circuits}\end{array}\right\}$ of size s on n variables

S.L.P. $= \quad Y_1, \ldots, Y_n, Y_{n+1}, \ldots, Y_s$

$\uparrow \qquad\qquad \uparrow$

$X_1, \ldots, X_n$

for each $i = n+1, \ldots, s$

there are j, k, really $j = j(i)$

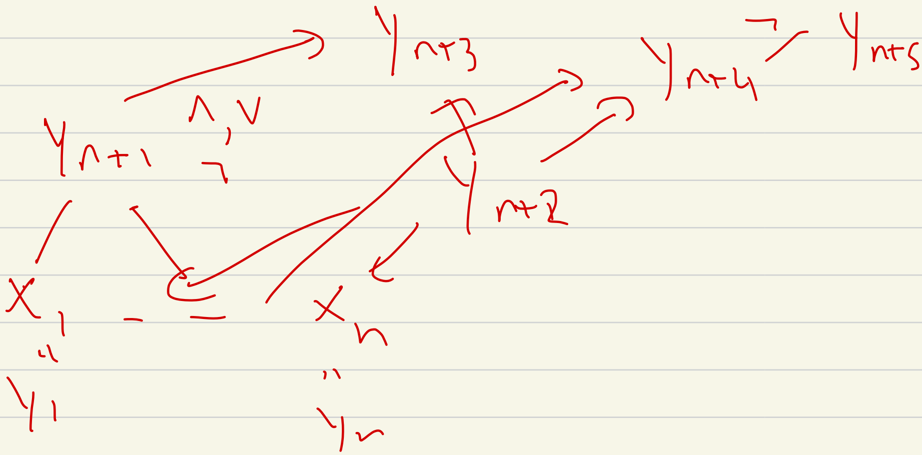
$k = k(i)$

s.t. either

$$Y_i = \begin{cases} Y_j \text{ AND } Y_k \\ Y_j \text{ OR } Y_k \\ \text{NEG } Y_j \end{cases}$$

S.L.P = Straight line programs

using Boolean logic, $\neg, \wedge, \vee$



Similarly show

$$\begin{pmatrix} \#\text{ S.L.P.} \\ (\#\text{ circuits}) \end{pmatrix} \begin{pmatrix} \text{size } n \\ s\text{, variables} \end{pmatrix}$$

$$< s^{2s} \begin{pmatrix} \text{less important terms} \end{pmatrix}$$

$$\log(s^{2s}) \sim 2^n \cdot c$$

same idea sets

$$2s \log_2 s \sim n + \log_2 c$$

---

① $n^s \simeq 2^n$, $s = \dfrac{2^n}{\log n}$

$s \log n \simeq 2^n$

② $s^s \simeq 2^n$, $s \simeq \dfrac{2^n}{n}$

③ $s^{2s} \sim 2^n$, $s^s \sim 2^{n/2}$

(3) $S \approx \dfrac{2^n}{2n}$

Claim:

$$S(n) = \left(\frac{2^n}{2n}\right) \cdot c$$

$c < 1$

$$S(n)^{2S(n)} \sim 2^n \cdot c$$

Upshot:

If $S(n) \approx \dfrac{2^n \cdot c}{\log n}$

$\log(\#\text{ formulas}(\text{size } S, n \text{ variables})) \sim 2^n \cdot c$

$$\log\left(\#\text{circuits}\begin{pmatrix}\text{size } n \\ s, \text{ variables}\end{pmatrix}\right) \sim 2^n c$$

main term

$$\log\left(s^{2s}\right)$$

if

$$s(n) = \frac{2^n}{2n} \cdot c$$

$\Rightarrow$

Thm (Shannon) The number of circuits size $\frac{2^n}{2n} \cdot c$ (for any $c < 1$) on n Boolean variables is $2^{\left[2^n \cdot c \left(1+o(1)\right)\right]}$

So for $c<1$, the number of Boolean functions on n-variables described (computed) by a circuit size $\leq \boxed{\frac{2^n}{2n} \cdot c}$

is $2^{2^n \cdot f(n)}$

where $f(n) \to c$ as $n \to \infty$.

So most Boolean functions on n variables

(1) are not described by circuits of size $\leq \boxed{\frac{2^n}{2n}(0.99)}$

(2) are not described by formulas of size $\boxed{\leq \frac{2^n}{\log n}(0.99)}$

Next time:

① Where are we today (2022)

in terms best lower bounds

for circuit & formula size

② Circuit & formula depth

Class ends
