CoCo (circuits)

amir yehudayoff

Turing machines

definition of algorithm / computer

hard to argue about: finite object defines infinite object

boolean circuits

definition of “chip”

seems easier to argue about: finite object defines a finite object

boolean circuits

definition: straight line program

$x_1, \ldots, x_n, 0, 1$ can be computed

if f, g are computed then $f \vee g, f \wedge g, \neg f$ can be computed

the cost grows by $+1$ in each step

boolean circuits

definition

a (boolean) circuit is a labelled dag

input gates labelled by $x_1, \ldots, x_n, 0, 1$

inner gates are $\vee, \wedge, \neg$ with fan-in ≤ 2

boolean circuits

definition

a (boolean) circuit is a labelled dag

input gates labelled by $x_1, \ldots, x_n, 0, 1$

inner gates are $\vee, \wedge, \neg$ with fan-in ≤ 2

remarks

if tree then called a formula

can have multiple outputs

boolean circuits

computation

every circuit C computes $\{0,1\}^n \to \{0,1\}$

costs

size(C) is number of vertices

depth(C) is length of longest directed path

complexity

universality

every $f : \{0,1\}^n \to \{0,1\}$ has a circuit

complexity

every f has complexities

$$\min\{\mathit{size}(C) : C \equiv f\}$$
$$\min\{\mathit{depth}(C) : C \equiv f\} = \min\{\mathit{depth}(F) : F \equiv f\}$$
$$\min\{\mathit{size}(F) : F \equiv f\}$$

C is a circuit and F a formula

Turing machines

TMs compute languages

need families of circuits $\{C_n\}_{n=1}^{\infty}$

$\{C_n\}$ computes L if for every n and $x \in \{0,1\}^n$

$$x \in L \quad \Leftrightarrow \quad C_n(x) = 1$$

languages and circuits

for $S : \mathbb{N} \to \mathbb{N}$ the class *size*$(S(n))$ comprises all $L \subseteq \{0,1\}^*$ such that there is $\{C_n\}$ for L with

$$|C_n| \leq O(S(n))$$

definition

$$P/poly = \bigcup_k size(n^k)$$

(justify name later)

circuits are strong

theorem

$$P \subsetneq P/poly$$

proof sketch

1. $\neq$

circuits are strong

theorem

$$P \subsetneq P/poly$$

proof sketch

1. $\neq$ undecidable

circuits are strong

theorem

$$P \subsetneq P/poly$$

proof sketch

1. $\neq$ undecidable

2. $\subseteq$ Cook-Levin proof ...

$P \subset P/poly$

for $L \in P$ there is a TM M for L with $TIME_M(n) = T \leq poly(n)$

$z_t \in \{0,1\}^B$ encodes state of computation at time t where $B \leq O(T+n)$

there is a circuit C with B inputs and B outputs such that

$$z_{t+1} = C(z_t) \quad \text{and} \quad size(C) \leq poly(B)$$

look at relevant output in $C^T(z_0) = C \circ C \circ \cdots \circ C(z_0)$

revisit $3SAT$ is NP-complete

by above $CIRCUIT\text{-}SAT$ is NP-complete

$CIRCUIT\text{-}SAT \leq_p 3SAT$

circuit C
each gate v computes f_v
variables y_v
3CNF formula checks that $y_v = f_v$ e.g.

$$v = u \wedge w \iff y_v = y_u \wedge y_w$$

advice

definition

for $T, A : \mathbb{N} \to \mathbb{N}$ the class

$$\mathit{TIME}(T(n))/A(n)$$

comprises all $L \subseteq \{0,1\}^*$ such that there is a TM M such that

$$\mathit{TIME}_M(n) \leq T(n)$$

and for every n, there is $a \in \{0,1\}^{A(n)}$ such that

$$x \in L \quad \Longleftrightarrow \quad M(x,a) = 1$$

equivalent definition

theorem

$$P/poly = \bigcup_k TIME(n^k)/n^k$$

explains the notation

equivalent definition

theorem

$$P/poly = \bigcup_k TIME(n^k)/n^k$$

sketch

$\subseteq$ the advice is the circuit

$\supseteq$ circuits simulate computations even with advice

uniformity

TMs are “uniform” computation (does not depend on n)

circuits or advices are “non-uniform” computation

what if?

$P \subset P/poly$—what about $NP \subset P/poly$?

can many witnesses be replaced by one advice?

what if?

$P \subset P/poly$—what about $NP \subset P/poly$?

can many witnesses be replaced by one advice?

theorem **[Karp-Lipton]**

if $NP \subset P/poly$ then $PH = \Sigma_2^p$

what if?

$P \subset P/poly$—what about $NP \subset P/poly$?

can many witnesses be replaced by one advice?

theorem **[Karp-Lipton]**

if $NP \subset P/poly$ then $PH = \Sigma_2^p$

remark suffices to show Π_2-SAT is in Σ_2^p

if $NP \subset P/poly$ then Π_2-SAT in Σ_2^p

an input to Π_2-SAT is

$$\forall x \in \{0,1\}^n \exists y \in \{0,1\}^n \varphi(x,y)$$

if $NP \subset P/poly$ then Π_2-SAT in Σ_2^p

an input to Π_2-SAT is

$$\forall x \in \{0,1\}^n \exists y \in \{0,1\}^n \varphi(x,y)$$

if $NP \subset P/poly$ then there is poly-size circuit C such that

$$\forall \varphi, x \quad \big(1 \equiv \exists y\ \varphi(x,y)\big) \iff \big(C(\varphi,x) = 1\big)$$

if $NP \subset P/poly$ then Π_2-SAT in Σ_2^p

an input to Π_2-SAT is

$$\forall x \in \{0,1\}^n \exists y \in \{0,1\}^n \varphi(x,y)$$

if $NP \subset P/poly$ then there is poly-size circuit C such that

$$\forall \varphi, x \quad \big(1 \equiv \exists y\ \varphi(x,y)\big) \iff \big(C(\varphi,x) = 1\big)$$

reducing search to decision there is poly-sized C_* such that

$$\forall \varphi, x \quad \big(1 \equiv \exists y\ \varphi(x,y)\big) \iff \big(\varphi(x,y_*) = 1 \text{ where } y_* = C_*(\varphi,x)\big)$$

if $NP \subset P/poly$ then Π_2-SAT in Σ_2^p

an input to Π_2-SAT is

$$\forall x \in \{0,1\}^n \exists y \in \{0,1\}^n \varphi(x,y)$$

if $NP \subset P/poly$ then there is poly-size circuit C such that

$$\forall \varphi, x \quad \big(1 \equiv \exists y\ \varphi(x,y)\big) \iff \big(C(\varphi,x) = 1\big)$$

reducing search to decision there is poly-sized C_* such that

$$\forall \varphi, x \quad \big(1 \equiv \exists y\ \varphi(x,y)\big) \iff \big(\varphi(x,y_*) = 1 \text{ where } y_* = C_*(\varphi,x)\big)$$

stated differently

$$\big(1 \equiv \forall x \exists y \varphi(x,y)\big) \iff \big(1 \equiv \exists C_* \forall x \varphi(x, C_*(\varphi,x))\big)$$

lower bounds

because $P \subset P/poly$

$$NP \not\subset P/poly \quad \Rightarrow P \neq NP$$

we “just” need to show that SAT can not be solved by poly-sized circuits

lower bounds

because $P \subset P/poly$

$$NP \not\subset P/poly \quad \Rightarrow P \neq NP$$

we "just" need to show that SAT can not be solved by poly-sized circuits

"good" news [Shannon]

there is $f : \{0,1\}^n \to \{0,1\}$ that require circuits of size $\geq \frac{2^n}{10n}$

counting arguments

there are hard functions

number of n-variate circuits of size $s > n$ is at most s^{3s}
think straight line program

number of n-variate functions 2^{2^n}

this is sharp

theorem **[Lupanov]**

every $f : \{0,1\}^n \to \{0,1\}$ has circuit-size $\leq \frac{10 \cdot 2^n}{n}$

this is sharp

theorem [Lupanov]

every $f : \{0,1\}^n \to \{0,1\}$ has circuit-size $\leq \frac{10 \cdot 2^n}{n}$

sketch

functions in k variables have circuit size $\approx 2^k$

compute "table" with all 2^{2^k} functions in k variables

read first $n-k$ variables and go to table

size is $\approx 2^{n-k} + 2^{2^k} \approx \frac{2^n}{n}$ for $k = \log(n - \log n)$

summary

circuits and formulas

advice and non-uniformity

lower bounds

counting arguments