Oliver Korten: The Hardest Explicit Construction

Petr Chmel

Definition 1 (Circuit).
A circuit $C : \{0,1\}^n \to \{0,1\}^m$ is a directed acyclic graph with n (ordered) nodes with indegree 0 and m (ordered) nodes with outdegree 0. All internal nodes (often called *gates*) are labeled by one of $\vee, \wedge, \neg$, with the $\neg$-gates having indegree (fan-in) 1, and $\vee, \wedge$ having fan-in 2. C computes a function by taking the input, evaluating the input nodes using the input bits, and then proceeding layer-by-layer until all output nodes have their evaluation.
The size of C, denoted by $|C|$, is the number of gates of C (we do not count the input and the output nodes).

Definition 2 (Empty and APEPP).
The problem Empty is a search problem defined as follows: given a circuit $C : \{0,1\}^n \to \{0,1\}^m$ with $m > n$, find an m-bit string outside the range of C.
The class APEPP is the class of all search problems that are reducible to Empty in polynomial time.

Observation 1 (Trivial algorithms for Empty).
Empty $\in \mathrm{TF}\Sigma_2^P$, and also Empty $\in \mathrm{FZPP}^{\mathrm{NP}}$.

Lemma 1 (Encoding of low-weight strings with fixed weight).
For any $k \leq n$, there exists a map $\Phi : \{0,1\}^{\log \binom{n}{k}} \to \{0,1\}^n$ computable in $\mathrm{poly}(n)$ time such that any n-bit string of weight k is in the range of Φ.

Corollary 1 (General encoding of low-weight strings).
For any $0 < \varepsilon < \frac{1}{2}$, there exists a map $\Phi : \{n - \varepsilon^2 n + \log(n)\} \to \{0,1\}$ computable in $\mathrm{poly}(n)$ time such that any n-bit string of weight at most $(\frac{1}{2} - \varepsilon)n$ is in range of Φ.

Definition 3 (Circuit complexity and Hard Truth Table).
Given a string x of length N, we say that x is computed by a circuit of size s, if there exists a circuit with $\lceil \log N \rceil$ inputs and s gates such that $C(i) = x_i$ for all $0 \leq i < |x|$. (If N is not a power of two, we do not care about $C(i)$ for $i \geq |x|$.)
Hard Truth Table is the following search problem: given 1^N, output a string x of length N such that x is not computed by any circuit of size at most $\frac{N}{2 \log N}$.

Definition 4 (Pseudorandom generator as a sequence and PRG).
A sequence $R = (x_1, \ldots, x_m)$ of n-bit strings is a pseudorandom generator if, for all circuits $C : \{0,1\}^n \to \{0,1\}$ of size n, $|\Pr_{x \sim R}[C(x) = 1] - \Pr_{y \sim \{0,1\}^n}[C(y) = 1]| \leq 1/n$.
PRG is the following search problem: given 1^n, output a pseudorandom generator $R = (x_1, \ldots, x_m)$, where all $x_i \in \{0,1\}^n$ (and $m = \mathrm{poly}(n)$).

Definition 5 ((k,ε)-extractor and (k,ε)-Extractor).
A function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is a (k,ε)-extractor, if for any two sets $X, Y \subseteq \{0,1\}^n$ of size 2^k, $|\Pr_{x \sim X, y \sim Y}[f(x,y) = 1] - \frac{1}{2}| \leq \varepsilon$.
For a pair of functions $k, \varepsilon : \mathbb{N} \to \mathbb{N}$, (k,ε)-Extractor is the following search problem: given 1^n, output a circuit C with $2n$ inputs such that the function $f_C : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ defined by C is a $(k(n), \varepsilon(n))$-Extractor.

Definition 6 (Rigid matrix and (ε, q)-Rigid).
A matrix $M \in \mathbb{F}_q^{n \times n}$ is (r,s)-rigid, if for any matrix $S \in \mathbb{F}_q^{n \times n}$ with at most s non-zero entries, $M + S$ has rank greater than r.
For any $q : \mathbb{N} \to \mathbb{N}$ such that $\forall n, q(n)$ is a prime power, (ε, q)-Rigid is the following search problem: given 1^n, output a matrix $M \in \mathbb{F}_{q(n)}^{n \times n}$ that is $(\varepsilon n, \varepsilon n^2)$-rigid.

Definition 7 (K^t Kolmogorov complexity and K_U^t-Random).
Let U be any fixed Turing machine, and let $t : \mathbb{N} \to \mathbb{N}$ be a time bound. For a string x, $K_U^t(x)$ is the length of the smallest string y such that U outputs x on the input y in $t(|x|)$ steps.
For a Turing machine U and a time bound t, K_U^t-Random is the following search problem: given 1^n, output an n-bit string x such that $K_U^t(x) \geq n - 1$.

Theorem 1 (Explicit construction problems)**.**
The following problems all reduce in polynomial time to Empty:

- Hard Truth Table,
- PRG,
- $(\log(n) + 2\log(1/\varepsilon(n)) + 3, \varepsilon(n))$-Extractor (for suitable efficiently computable $\varepsilon(n)$),
- Explicit construction of strongly explicit Ramsey graphs,
- (ε, q)-Rigid for $\varepsilon \leq \frac{1}{16}$ and any suitable efficiently computable $q(n)$,
- K_U^t-Random.

Definition 8 (Circuit base and inverter reduction)**.**
A basis $\mathcal{C}$ is a (possibly infinite) set of boolean functions such as $\{\wedge, \vee, \neg\}$. A $\mathcal{C}$-circuit is a circuit in which all gates are labeled by one of the functions from $\mathcal{C}$.
A basis is *sufficiently strong* if it can compute the two-input functions $\wedge, \vee$, and the one-input $\neg$ with constantly many gates.
For a basis $\mathcal{C}$, a $\mathcal{C}$-inverter oracle is an oracle, that, given a $\mathcal{C}$-circuit C and a string y, determines whether there exists an x such that $C(x) = y$ and produces such x if it exists. A $\mathcal{C}$-inverter reduction is a polynomial time reduction that uses a $\mathcal{C}$-inverter oracle.

Definition 9 (Generalized Empty and APEPP)**.**
Given a basis $\mathcal{C}$ and a strictly increasing function $f : \mathbb{N} \to \mathbb{N}$, we define the search problem $\text{Empty}^{\mathcal{C}}_{f(n)}$ as follows: given a $\mathcal{C}$-circuit C with n input wires and $f(n)$ output wires, find an $f(n)$-bit string that is not in the range of C. If the subscript is missing, any circuit with more output bits than input bits is allowed.
The class $\text{APEPP}^{\mathcal{C}}$ is the class of all search problems that are reducible to $\text{Empty}^{\mathcal{C}}$ in polynomial time.

Lemma 2 (Fixed output length is still complete)**.**
For any basis $\mathcal{C}$, $\text{Empty}^{\mathcal{C}}_{2n}$ is complete for $\text{APEPP}^{\mathcal{C}}$ under $\mathcal{C}$-inverter reductions.

Definition 10 (ε-$\text{Hard}^{\mathcal{C}}$)**.**
We define the search problem ε-$\text{Hard}^{\mathcal{C}}$ as follows: given 1^N, output a string x of length N such that x cannot be computed by $\mathcal{C}$-circuits of size N^{ε}.

Theorem 2 (General reduction from Empty to Hard)**.**
For a sufficiently strong basis $\mathcal{C}$ and a constant $\varepsilon > 0$ such that ε-$\text{Hard}^{\mathcal{C}}$ is total for sufficiently large input lengths, $\text{Empty}^{\mathcal{C}}$ reduces to ε-$\text{Hard}^{\mathcal{C}}$ under $\mathcal{C}$-inverter reductions.

Corollary 2 (The hardest explicit construction)**.**
For any $0 < \varepsilon < 1$, ε-Hard is complete for APEPP under P^{NP} reductions.

Theorem 3 (Lower bounds vs algorithms)**.**
There exists a language in E^{NP} with circuit complexity $2^{\Omega(n)}$ if and only if there is a P^{NP} algorithm for Empty.

Corollary 3 (Worst-case to worst-case hardness amplification for E^{NP})**.**
If there is a language in E^{NP} with circuit complexity $2^{\Omega(n)}$, then there is a language in E^{NP} requiring circuits of size $\frac{2^n}{2n}$.

Corollary 4 (Worst-case to worst-case hardness amplification for EXP^{NP})**.**
If there is a language in EXP^{NP} with circuit complexity $2^{n^{\Omega(1)}}$, then there is a language in EXP^{NP} requiring circuits of size $\frac{2^n}{2n}$.