Oliver Korten: The Hardest Explicit Construction (Informally)

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 from n-bit strings to m-bit strings with $m > n$, find a string that cannot be the output of the circuit.
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 is a search problem that always has a solution, and the solution can be verified in coNP (or with an NP oracle).
We can solve Empty by randomly taking an m-bit output string and using an oracle to check if it is in the range or not. As at least 1/2 of the strings are outside the range, we will end this in polynomial time with high probability.

Lemma 1 (Encoding of low-weight strings with fixed weight)**.**
For a fixed $k \leq n$, there is a poly-time computable function that has all n-bit strings with precisely k ones in its range.

Corollary 1 (General encoding of low-weight strings)**.**
For any $0 < \varepsilon < \frac{1}{2}$, there is a poly-time computable function that has all n-bit strings of weight at most $(\frac{1}{2} - \varepsilon)n$ in its range.

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 $\frac{N}{2 \log N}$.

Theorem 1 (Explicit construction problems)**.**
If we can solve Empty, we can solve the following problems with polynomial-time overhead:

- constructing truth tables with high circuit complexity,
- constructing (complexity-theoretic) pseudorandom generators,
- constructing randomness extractors,
- constructing strongly explicit Ramsey graphs,
- constructing rigid matrices,
- constructing time-bounded Kolmogorov random strings.

Definition 8 (Circuit base and inverter reduction)**.**
A basis $\mathcal{C}$ is a set of boolean functions. If we use the functions in $\mathcal{C}$ to label the gates, we call the circuit a $\mathcal{C}$-circuit.
A basis is *sufficiently strong* if it can compute the AND of two bits, the OR of two bits, and the negation of one bit 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 its output either says "the output is out of range" or returns an input that C evaluates to the output. A $\mathcal{C}$-inverter reduction is a poly-time reduction that uses a $\mathcal{C}$-inverter oracle.

Definition 9 (Generalized EMPTY and APEPP)**.**
We extend EMPTY to $\text{EMPTY}^{\mathcal{C}}_{f(n)}$ by adding two parameters: instead of "usual circuits", we work with $\mathcal{C}$-circuits, and we now require that the circuit from n-bit strings outputs $f(n)$-bit strings. 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 there are languages that have a truth table hard enough for all N large enough (and thus ε-$\text{HARD}^{\mathcal{C}}$ has a solution for all N large enough), $\text{EMPTY}^{\mathcal{C}}$ reduces to ε-$\text{HARD}^{\mathcal{C}}$ under $\mathcal{C}$-inverter reductions.

Corollary 2 (The hardest explicit construction)**.**
For any $0 < \varepsilon < 1$, solving one of ε-HARD and EMPTY implies the ability to solve the other with a P^{NP} overhead.

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 $\frac{2^n}{2n}$.