On the Journey from NP hardness to TFNP

Akshat Yaparla, Mark Chen

8 February 2024

TFNP in Impagliazzo's Worlds

TFNP in Impagliazzo's Worlds

- This paper discusses the effect of average-case hardness in NP on average-case hardness (!!) of TFNP.

TFNP in Impagliazzo's Worlds

- This paper discusses the effect of average-case hardness in NP on average-case hardness (!!) of TFNP.
- Average-case NP hardness puts us in at least *Pessiland*.

TFNP in Impagliazzo's Worlds

- This paper discusses the effect of average-case hardness in NP on average-case hardness (!!) of TFNP.
- Average-case NP hardness puts us in at least *Pessiland*.
- However, theorems in this paper do not depend on the existence of one-way functions.

TFNP in Impagliazzo's Worlds

- This paper discusses the effect of average-case hardness in NP on average-case hardness (!!) of TFNP.
- Average-case NP hardness puts us in at least *Pessiland*.
- However, theorems in this paper do not depend on the existence of one-way functions.
  - This puts us in an intermediate area between Pessiland and Minicrypt, where additional assumptions such as NW assumptions were made on top of Pessiland assumptions but such assumptions are independent from OWFs.

Theorem 1

Theorem 1

- **Theorem 1.** [1] If there exists a hard-on-average language in NP, then there exists a hard-on-average problem in *nonuniform* TFNP (i.e. TFNP/poly).

Theorem 1

- **Theorem 1.** [1] If there exists a hard-on-average language in NP, then there exists a hard-on-average problem in *nonuniform* TFNP (i.e. TFNP/poly).
  - What does "*nonuniform* TFNP (i.e. TFNP/poly)" mean? It means that the verifier for this problem is allowed to run a different algorithm for each input length (or get advice depending on the input length).

Assumptions

Here, we summarize the assumptions and explain how they are related:

- (**Theorem 1 Assumption** - Pessiland): Hard-on-average language exists in NP.
- (**Assumption 1** - Not Quite Minicrypt): There exists function with deterministic time complexity $2^{O(n)}$ and Π_2 circuit complexity $2^{\omega(n)}$ (NW assumptions).
- (**Assumption 2** - Minicrypt): Injective OWF (iOWF) and non-interactive witness-indistinguishable proof systems (Zaps) exist.

* More information on these primitives and assumptions is in the background document we prepared (and in the paper). We will focus on Theorem 1.

Assumptions

- (Theorem 1 Assumption + Assumption 1) $\implies$ **Corollary.** There exists a hard-on-average language in *uniform* TFNP.
- (Theorem 1 Assumption + Assumption 1&2) $\implies$ **Theorem 3.** There exists a hard-on-average problem in TFNP such that any instance has at most *two* solutions.

* More information on these primitives and assumptions is in the background document we prepared (and in the paper). We will focus on Theorem 1.

Diagram of Results

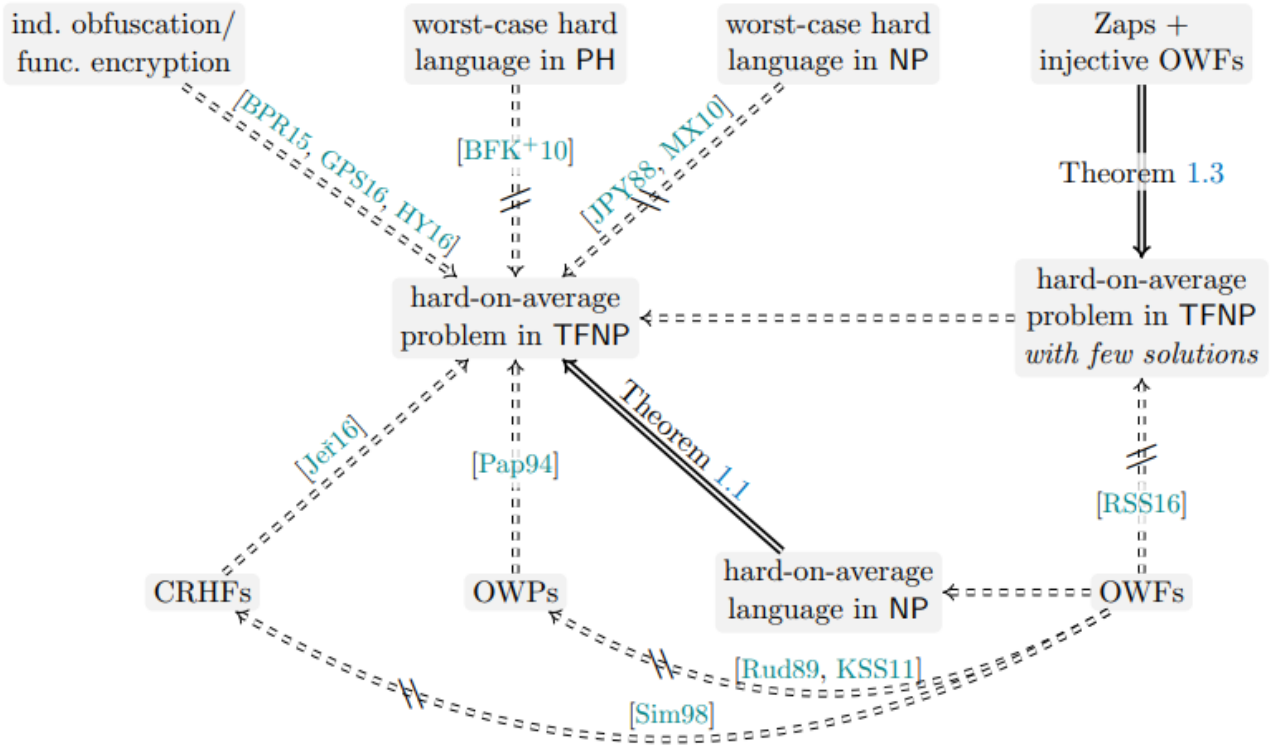

Figure: Diagram of results in the paper.

Average Case Complexity

Definition. A language L is said to be *hard-on-average* for NP over distribution $\mathcal{D}(1^n) \in \{0,1\}^n$ for any probabilistic polynomial time algorithm A, there exists a negligible function $\epsilon(\cdot)$ such that for all large enough n,

$$\mathbf{Pr}_{A,x \leftarrow \mathcal{D}_n(r)}[A(x) = L(x)] \leq \frac{1}{2} + \epsilon(n)$$

Average Case Complexity

Definition. A language L is said to be *hard-on-average* for NP over distribution $\mathcal{D}(1^n) \in \{0,1\}^n$ for any probabilistic polynomial time algorithm A, there exists a negligible function $\epsilon(\cdot)$ such that for all large enough n,

$$\mathbf{Pr}_{A,x \leftarrow \mathcal{D}_n(r)}[A(x) = L(x)] \leq \frac{1}{2} + \epsilon(n)$$

- If $\mathcal{D}_n$ is a uniform distribution, then we simply state L as being hard-on-average for NP.

Average Case Complexity

Definition. A language L is said to be *hard-on-average* for NP over distribution $\mathcal{D}(1^n) \in \{0,1\}^n$ for any probabilistic polynomial time algorithm A, there exists a negligible function $\epsilon(\cdot)$ such that for all large enough n,

$$\mathbf{Pr}_{A,x \leftarrow \mathcal{D}_n(r)}[A(x) = L(x)] \leq \frac{1}{2} + \epsilon(n)$$

- If $\mathcal{D}_n$ is a uniform distribution, then we simply state L as being hard-on-average for NP.
- Otherwise, we must specify that L is hard-on-average for nonuniform NP.

Proof of Theorem 1 - What and Where is Non-uniformity

Before we dive into the proof, it is critical to note the difference between two kinds of non-uniformity.

Proof of Theorem 1 - What and Where is Non-uniformity

Before we dive into the proof, it is critical to note the difference between two kinds of non-uniformity.

- **(Language L or relation R)** Nonuniformity means that the algorithms are nonuniform. For example, in circuit complexity, different algorithms may be run on input strings of different lengths (that is, the same algorithm is run on two strings of the same length; different algorithms may be run on two strings of different lengths).
  Notation: $\mathcal{D}(1^n) \to$ instance of L / R of length n.

Proof of Theorem 1 - What and Where is Non-uniformity

Before we dive into the proof, it is critical to note the difference between two kinds of non-uniformity.

(i) **(Language L or relation R)** Nonuniformity means that the algorithms are nonuniform. For example, in circuit complexity, different algorithms may be run on input strings of different lengths (that is, the same algorithm is run on two strings of the same length; different algorithms may be run on two strings of different lengths).
Notation: $\mathcal{D}(1^n) \to$ instance of L / R of length n.

(ii) **(Verifier)** Nonuniformity means the distribution of witness strings are nonuniform.
Notation: $\mathcal{D}_n(1^n) \equiv \mathcal{D}_n(1^n, \mathcal{S}_n) \to y \in \{0,1\}^n$.

Proof of Theorem 1

Proof of Theorem 1

- We want to show for a fixed input n that any hard problem L over distribution $\mathcal{D}$ also implies a hard distributional search problem, corresponding to FNP.

Proof of Theorem 1

- We want to show for a fixed input n that any hard problem L over distribution $\mathcal{D}$ also implies a hard distributional search problem, corresponding to FNP.
- Then, we extend this over to total problems to prove hard-on-average for nonuniform TFNP.

Proof of Theorem 1

Construct a relation R_L that is the corresponding search problem for a hard language $L \in \text{NP}$.

Proof of Theorem 1

Construct a relation R_L that is the corresponding search problem for a hard language $L \in \mathrm{NP}$.

- First, pick an $x \leftarrow \mathcal{D}(1^n)$ in order to find a w such that $R_L(x, w) = 1$ is hard.

Proof of Theorem 1

Construct a relation R_L that is the corresponding search problem for a hard language $L \in \text{NP}$.

- First, pick an $x \leftarrow \mathcal{D}(1^n)$ in order to find a w such that $R_L(x, w) = 1$ is hard.
- Let $p(\cdot)$ be a polynomial such that $\mathbf{Pr}_{x \leftarrow \mathcal{D}(1^n)}[L(x) = 1] > p(n)$. Otherwise, a "no" algorithm could distinguish L and contradict $\mathcal{D}$ being a hard distribution.

Proof of Theorem 1

Construct a relation R_L that is the corresponding search problem for a hard language $L \in \text{NP}$.

- First, pick an $x \leftarrow \mathcal{D}(1^n)$ in order to find a w such that $R_L(x, w) = 1$ is hard.
- Let $p(\cdot)$ be a polynomial such that $\mathbf{Pr}_{x \leftarrow \mathcal{D}(1^n)}[L(x) = 1] > p(n)$. Otherwise, a "no" algorithm could distinguish L and contradict $\mathcal{D}$ being a hard distribution.
- Define a set of $k = n^2 \cdot p(n)$ strings enumerated $s_i \in \{0,1\}^n$. Then, the problem $L_{s_1 \ldots s_k}$ under a new uniform distribution $\mathcal{D}'$ is defined by

$$R_{s_1 \ldots s_k}(r, w) = \bigvee_{i \in [k]} R_L(x_i, w), x \leftarrow \mathcal{D}(1^n)(r \oplus s_i)$$

Proof of Theorem 1

Lemma. Let $(L, \mathcal{D})$ be a hard distributional search problem. Let $(L', \mathcal{D}')$ be a distributional serach problem related to $(L, \mathcal{D})$ that satisfies the following conditions:

Proof of Theorem 1

Lemma. Let $(L, \mathcal{D})$ be a hard distributional search problem. Let $(L', \mathcal{D}')$ be a distributional serach problem related to $(L, \mathcal{D})$ that satisfies the following conditions:

1. L' is an "OR" form, meaning that there exists efficiently computable functions $f_1, \ldots, f_k$ such that $R_{L'}(x', w) = \bigvee_{i \in [k]} R_L(f_i(x'), w)$ where k is some polynomially bounded function of n.

Proof of Theorem 1

Lemma. Let $(L, \mathcal{D})$ be a hard distributional search problem. Let $(L', \mathcal{D}')$ be a distributional serach problem related to $(L, \mathcal{D})$ that satisfies the following conditions:

1. L' is an "OR" form, meaning that there exists efficiently computable functions $f_1, \ldots, f_k$ such that $R_{L'}(x', w) = \bigvee_{i \in [k]} R_L(f_i(x'), w)$ where k is some polynomially bounded function of n.
2. For every $i \in [k]$, the marginal distribution of $f_i(x')$ under $x' \leftarrow \mathcal{D}'$ is identical to the distribution of $x \leftarrow \mathcal{D}(1^n)$.

Proof of Theorem 1

Lemma. Let $(L, \mathcal{D})$ be a hard distributional search problem. Let $(L', \mathcal{D}')$ be a distributional serach problem related to $(L, \mathcal{D})$ that satisfies the following conditions:

1. L' is an "OR" form, meaning that there exists efficiently computable functions $f_1, \ldots, f_k$ such that $R_{L'}(x', w) = \bigvee_{i \in [k]} R_L(f_i(x'), w)$ where k is some polynomially bounded function of n.
2. For every $i \in [k]$, the marginal distribution of $f_i(x')$ under $x' \leftarrow \mathcal{D}'$ is identical to the distribution of $x \leftarrow \mathcal{D}(1^n)$.
3. For any fixed instance $x^* \leftarrow \mathcal{D}'(1^n)$ conditioned on $f_i(x') = x^*$ is efficiently samplable (given r).

Proof of Theorem 1

Proof Sketch. Just want to follow a reduction by reducing the original distributional search problem $(L, \mathcal{D})$ to the new problem, implying the new problem $(L', \mathcal{D}')$ is also a hard distributional search problem.

- Note that our problem $(L_{s_1,\ldots,s_k}, \mathcal{D}')$ satisfies the conditions of the previous lemma.

Proof of Theorem 1

We also want $L_{s_1,\ldots,s_k}$ to be a total language to satisfy hardness for nonuniform TFNP. That is,

$$\forall r \in \{0,1\}^n : L_{s_1,\ldots,s_k}(r) = 1.$$

The authors claim that

$$\mathbf{Pr}_{s_1,\ldots,s_k}[L_{s_1,\ldots,s_k} \textit{ is total}] \geq \frac{3}{4}$$

Proof of Theorem 1

Proof. Fix any string r. Picking an s at random yields

$$\mathbf{Pr}_{s \leftarrow \{0,1\}^{kn}, x \leftarrow \mathcal{D}(1^n; r \oplus s)}[L(x) = 1] \geq 1/p(n),$$

since for any fixed r, the string $r \oplus s$ is chosen at uniformly random. Now, utilize that for any s_i, the event is independent, and it holds that for any fixed r and $k = n^2 \cdot p(n)$,

$$\mathbf{Pr}_{s \leftarrow \{0,1\}^{kn}, x \leftarrow \mathcal{D}(1^n; r \oplus s)}[\forall i : L(x_i) = 0] \leq (1 - 1/p(n))^k \leq 2^{-n^2},$$

Union bounding over all $r \in \{0,1\}^n$ yields

$$\mathbf{Pr}_{s \leftarrow \{0,1\}^{kn}}[L_{s_i} \textit{ is not total}] \leq 2^{-n^2} \cdot 2^n \leq 1/4.$$

Finishing the proof.

Proof of Theorem 1

Proof of Theorem 1

- For any n, we fix k strings *non-uniformly* to get a total $L_{s_1,\ldots,s_k}$ to obtain language L^*.

Proof of Theorem 1

- For any n, we fix k strings *non-uniformly* to get a total $L_{s_1,\ldots,s_k}$ to obtain language L^*.
- This is hard-on-average search problem under uniform distribution D^*.

Proof of Theorem 1

- For any n, we fix k strings *non-uniformly* to get a total $L_{s_1,\ldots,s_k}$ to obtain language L^*.
- This is hard-on-average search problem under uniform distribution D^*.
- This problem is also total: for any n, every $r \in \{0,1\}^n$ must have a solution since we chose a total $L_{s_1,\ldots,s_k}$.

App.I - NW PRGs

Definition (Nisan-Wigderson (NW) PRGs). A function $G : \{0,1\}^{d(n)} \to \{0,1\}^n$ is an NW-type PRG against circuits of size $t(n)$ if

App.I - NW PRGs

Definition (Nisan-Wigderson (NW) PRGs). A function $G : \{0,1\}^{d(n)} \to \{0,1\}^n$ is an NW-type PRG against circuits of size $t(n)$ if

1. It is computable in time $2^{O(d(n))}$,

App.I - NW PRGs

Definition (Nisan-Wigderson (NW) PRGs). A function $G : \{0,1\}^{d(n)} \to \{0,1\}^n$ is an NW-type PRG against circuits of size $t(n)$ if

(i) It is computable in time $2^{O(d(n))}$,

(ii) Any circuit of size at most $t(n)$ distinguishes $U \leftarrow \{0,1\}^n$ from $G(s)$, where $s \leftarrow \{0,1\}^{d(n)}$, with advantage at most $1/t(n)$.

App.II - Cryptographic vs Complexity Theor. PRGs

Both stem out of a common definition. A PRG $G : \{0,1\}^{s(n)} \to \{0,1\}^n$ is a length expanding function such that for a negligible function $\epsilon(\cdot)$ and for any probabilistic polynomial time algorithm A,

$$\left|\mathbf{Pr}_A[A(U_n) = 1] - \mathbf{Pr}_A[A(G(U_{s(n)})) = 1]\right| \leq \epsilon(n)$$

We continue discussing how the two are different on the next slide.

App.II - Cryptographic vs Complexity Theor. PRGs

It is unknown if cryptographic PRGs are stronger than the complexity theoretic PRGs, or the other way around, because of the followng two degrees of freedoms.

App.II - Cryptographic vs Complexity Theor. PRGs

It is unknown if cryptographic PRGs are stronger than the complexity theoretic PRGs, or the other way around, because of the followng two degrees of freedoms.

- **(Stretch)** Recall $s(n) \leq n$. If $s(n) = \log(n)$, then the codomain is exponentially "stretched," which makes it harder to cheat adversaries. In comparison, $s(n) = n$ is an easier requirement. Complexity PRGs have larger stretches.

App.II - Cryptographic vs Complexity Theor. PRGs

It is unknown if cryptographic PRGs are stronger than the complexity theoretic PRGs, or the other way around, because of the followng two degrees of freedoms.

- **(Stretch)** Recall $s(n) \leq n$. If $s(n) = \log(n)$, then the codomain is exponentially "stretched," which makes it harder to cheat adversaries. In comparison, $s(n) = n$ is an easier requirement. Complexity PRGs have larger stretches.
- **(Adversaries Safe Against)** How complex of a circuit does the adversary need to at least be to differentiate the PRG from uniform distribution.
  Cryptographic PRGs are safe against harder adversaries.


Pavel Hubáček, Moni Naor, and Eylon Yogev.
The journey from np to tfnp hardness.