Introduction to Functional Classes and Range Avoidance

Nikolai Chukhin

28.11.2025

Content

Search vs. Decision

- P is a class of functions $L\colon \{0,1\}^* \to \{0,1\}$ computable in polynomial time.

Search vs. Decision

- P is a class of functions $L\colon \{0,1\}^* \to \{0,1\}$ computable in polynomial time.
- FP: given an input x, compute some y such that $R(x,y) = 1$ in time polynomial in $|x|$, whenever such y exists, where $R\colon \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$.

Brain Teaser: Provide a problem whose decision version lies in P, whereas the search version is not known to be in FP.

Search vs. Decision

- P is a class of functions $L\colon \{0,1\}^* \to \{0,1\}$ computable in polynomial time.
- FP: given an input x, compute some y such that $R(x,y) = 1$ in time polynomial in $|x|$, whenever such y exists, where $R\colon \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$.

Primality testing belongs to P, yet integer factorization is not known to be in FP.

Theorem ([BG94])

Suppose $\mathsf{EE} \neq \mathsf{NEE}$. *Then there is a language in* NP *which search version cannot be reduces to decision.*

From NP to FNP

Definition

FNP: polynomial-time verifiable relations $R(x, y)$, where $|y| \leq \mathsf{poly}(|x|)$.
Task: given x, find y with $R(x, y) = 1$ using nondeterminism or report none exists.

Brain Teaser: Consider the following relation: $R(F, x) = F(x)$ iff F is a CNF formula and x is an assignment. Does it belong to FNP?

From NP to FNP

Definition

FNP: polynomial-time verifiable relations $R(x, y)$, where $|y| \leq \mathsf{poly}(|x|)$.
Task: given x, find y with $R(x, y) = 1$ using nondeterminism or report none exists.

Brain Teaser: Consider the following relation: $R(F, x) = F(x)$ iff F is a CNF formula and x is an assignment. Does it belong to FNP?

Lemma

Finding a satisfying assignment of 3-SAT *is* FNP*-complete.*

Brain Teaser: What about adding $R(F, \bot)=1 \iff F$ is unsatisfiable? Does it belong to FNP?

When does search collapse to decision?

Theorem

$\mathsf{FP} = \mathsf{FNP} \iff \mathsf{P} = \mathsf{NP}$.

Proof idea

($\Rightarrow$) Let $L \in \mathsf{NP}$, then there is a balanced poly-time relation $R(x, y)$ such that

$$x \in L \iff \exists y \colon R(x, y).$$

Since $\mathsf{FP} = \mathsf{FNP}$, find such y and verify $R(x, y)$.

When does search collapse to decision?

Theorem

$\mathsf{FP} = \mathsf{FNP} \iff \mathsf{P} = \mathsf{NP}$.

Proof idea

($\Rightarrow$) Let $L \in \mathsf{NP}$, then there is a balanced poly-time relation $R(x, y)$ such that

$$x \in L \iff \exists y \colon R(x, y).$$

Since $\mathsf{FP} = \mathsf{FNP}$, find such y and verify $R(x, y)$.

($\Leftarrow$) Given a relation $R(x, y)$. Let

$$L = \{x \colon \exists y \, R(x, y)\} \in \mathsf{NP} = \mathsf{P}.$$

Brain Teaser: then?

When does search collapse to decision?

Theorem

$\mathsf{FP} = \mathsf{FNP} \iff \mathsf{P} = \mathsf{NP}$.

Proof idea

($\Rightarrow$) Let $L \in \mathsf{NP}$, then there is a balanced poly-time relation $R(x, y)$ such that

$$x \in L \iff \exists y \colon R(x, y).$$

Since $\mathsf{FP} = \mathsf{FNP}$, find such y and verify $R(x, y)$.

($\Leftarrow$) Given a relation $R(x, y)$. Let

$$L = \{x \colon \exists y \, R(x, y)\} \in \mathsf{NP} = \mathsf{P}.$$

Iteratively guess each bit of y.

Total search: TFNP

Definition ([MP91])

TFNP: $R \in \mathsf{FNP}$ such that $\forall x \, \exists y \, R(x, y)$.

Total search: TFNP

Definition ([MP91])

TFNP: $R \in \mathsf{FNP}$ such that $\forall x \, \exists y \, R(x, y)$.

Theorem (Informally, [MP91])

$\mathsf{TFNP} = \mathsf{F}(\mathsf{NP} \cap \mathsf{coNP})$, *where in* $\mathsf{F}(\mathsf{NP} \cap \mathsf{coNP})$ *we are given* x *and we need to find certificate of whether there is* y *such that* $R(x, y) = 1$.

Proof idea

Brain Teaser: Why $\mathsf{TFNP} \subseteq \mathsf{F}(\mathsf{NP} \cap \mathsf{coNP})$?

Total search: TFNP

Definition ([MP91])

TFNP: $R \in \mathsf{FNP}$ such that $\forall x \, \exists y \, R(x, y)$.

Theorem (Informally, [MP91])

$\mathsf{TFNP} = \mathsf{F}(\mathsf{NP} \cap \mathsf{coNP})$, *where in* $\mathsf{F}(\mathsf{NP} \cap \mathsf{coNP})$ *we are given* x *and we need to find certificate of whether there is* y *such that* $R(x, y) = 1$.

Proof idea

Clearly, $\mathsf{TFNP} \subseteq \mathsf{F}(\mathsf{NP} \cap \mathsf{coNP})$. In the other way, decompose R into $R_1 \cup R_2$ where

$$(x, 1y) \in R_1 \text{ if } R(x, y), \text{ and } (x, 0y) \in R_2 \text{ if } \neg R(x, y).$$

Then, $R_1 \cup R_2 \in \mathsf{TFNP}$.

Hard problems in TFNP

Theorem ([Meg88])

If some $R \in \mathsf{TFNP}$ is NP-hard under Karp-Levin reductions, then $\mathsf{NP} = \mathsf{coNP}$.

It is believed, that there is no complete problem in TFNP.

Theorem ([Pud15])

There exists an oracle A such that the class TFNP^A does not contain a complete problem under many-to-one reductions.

Content

Subclasses of TFNP

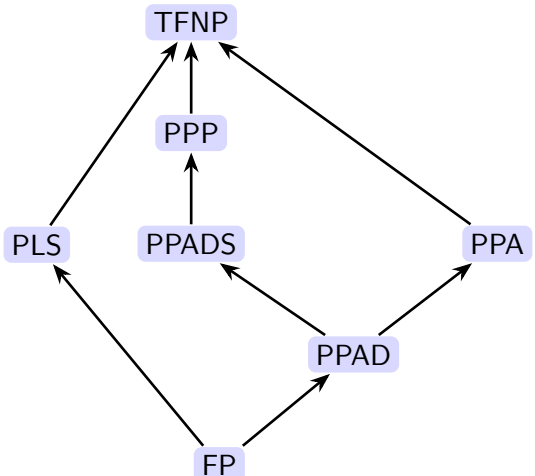

Figure: Classical TFNP subclasses. A directed arrow from class A to B means $A \subseteq B$ [LPR24].

PPP

Definition (Pigeon)

Given a circuit $C : \{0,1\}^n \to \{0,1\}^n$, output either x with $C(x) = 0^n$ or a collision $x \neq y$ with $C(x) = C(y)$.

Definition ([Pap94])

PPP: total relations polynomial-time reducible to PIGEON.

Brain Teaser: Any examples of problems in PPP?

PPP

Definition (Pigeon)

Given a circuit $C : \{0,1\}^n \to \{0,1\}^n$, output either x with $C(x) = 0^n$ or a collision $x \neq y$ with $C(x) = C(y)$.

Definition ([Pap94])

PPP: total relations polynomial-time reducible to PIGEON.

Brain Teaser: Any examples of problems in PPP?

- *Pigeonhole Equal Subset Sum* [WY92; Pap94]: find distinct $A, B \subseteq [n]$ with $\sum_{i \in A} w_i = \sum_{i \in B} w_i$ when $\sum_i w_i < 2^n - 1$[1].

[1] Conjectured to be PPP-complete [Pap94].

PPP and cryptography

- If PPP = FP, one-way permutations do not exist [Pap94].
- Integer factorization randomized-reduces to PPP; under GRH, deterministic reductions are known [Jer16]. It is unknown whether factoring is in PPP *unconditionally*.

Content

From collisions to avoidance

- PPP: *"if there are more pigeons than pigeonholes, there must be a pigeonhole with two or more pigeons"*
- *Dual PPP: "if there are more pigeonholes than pigeons, then there must be an empty pigeonhole"*

From collisions to avoidance

- PPP: *"if there are more pigeons than pigeonholes, there must be a pigeonhole with two or more pigeons"*
- *Dual PPP: "if there are more pigeonholes than pigeons, then there must be an empty pigeonhole"*

Definition (Empty [KKMP21])

Given $C : [N] \to [M]$, where $N < M$, find y not in the image of C.

Brain Teaser: What is the difference with Pigeon?

From collisions to avoidance

- PPP: *"if there are more pigeons than pigeonholes, there must be a pigeonhole with two or more pigeons"*
- *Dual PPP: "if there are more pigeonholes than pigeons, then there must be an empty pigeonhole"*

Definition (Empty [KKMP21])

Given $C : [N] \to [M]$, where $N < M$, find y not in the image of C.

We cannot easily put Empty in TFNP, since it needs two quantifiers! Informally, Empty $\in \mathsf{TF}\Sigma_2$.

Definition

PEPP: total relations which are polynomial time reducible to EMPTY.

FNP sits inside PEPP

Theorem ([KKMP21])

FNP $\subseteq$ PEPP.

Proof.

We prove that SAT can be reduced to EMPTY. Let ϕ be a CNF formula with n variables, w.l.o.g not satisfied by 1^n.
Consider the following circuit C from $[2^n - 1]$ to $[2^n]$:

- Given $t \in \{0,1\}^n \setminus \{1^n\}$, C tests whether t satisfies ϕ.
- If it does, then $C(t) = 1^n$, and if it does not then $C(t) = t$.

Now, if we could find a solution $s \in [2^n]$ not in the range of C, then we would have solved the SAT problem for ϕ: If $s \neq 1^n$ then ϕ is satisfiable and s satisfies it; otherwise, ϕ is unsatisfiable. □

Matrix Rigidity is in PEPP

Definition ([Val77])

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, $\text{rank}(M + S) \geq r$.
The ε-RIGID problem: given 1^n, output $M \in \mathbb{F}_q^{n \times n}$ which is $(\varepsilon n, \varepsilon n^2)$-rigid.

Theorem ([Val77])

Any $(\delta n, n^{1+\delta})$-rigid matrix cannot be computed by linear size, logarithmic depth arithmetic circuits.

Theorem ([KKMP21; Kor21])

*For any $\varepsilon \leq \frac{1}{16}$, the ε-*RIGID *problem reduces in polynomial time to* EMPTY*.*

Matrix Rigidity is in PEPP

Definition ([Val77])

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, $\operatorname{rank}(M + S) \geq r$.
The ε-RIGID problem: given 1^n, output $M \in \mathbb{F}_q^{n \times n}$ which is $(\varepsilon n, \varepsilon n^2)$-rigid.

Theorem ([KKMP21; Kor21])

For any $\varepsilon \leq \frac{1}{16}$, the ε-RIGID problem reduces in polynomial time to EMPTY.

Proof idea

Let $M \in \mathbb{F}_q^{n \times n}$, which is not (r, s)-rigid. Then, $M = L \cdot R + s$, where $L \in \mathbb{F}_q^{n \times r}, R \in \mathbb{F}_q^{r \times n}$ and S has at most s non-zero entries. Then, given L, R and S one can recover M. Hence, to encode M one needs at most

$$(2nr + s) \log q + \log \binom{n^2}{s} < n^2 \log q.$$

Hard Truth Table is in PEPP

Definition (Hard Truth Table problem)

Given 1^N, output a string x of length N such that x cannot be computed by any circuit of size at most $\frac{N}{2 \log N}$.

Theorem ([Kor21])

Hard Truth Table *reduces in polynomial time to* Empty.

Brain Teaser: Any ideas?

Hard Truth Table is in PEPP

Definition (Hard Truth Table problem)

Given 1^N, output a string x of length N such that x cannot be computed by any circuit of size at most $\frac{N}{2 \log N}$.

Theorem ([Kor21])

Hard Truth Table *reduces in polynomial time to* Empty.

Proof idea

The mapping Φ interprets its input as a circuit on $\lceil \log N \rceil$ bits, evaluates its output on every possible input to generate a $2^{\lceil \log N \rceil}$-bit truth table, and truncates this table to length exactly N.
Since circuits can be efficiently decoded, and many functions have circuit complexity exceeding $\frac{N}{2 \log N}$, there must exist an output that lies outside the range of Φ.

Abundant Avoidance

Definition

Let $\alpha\colon \mathbb{N} \to \mathbb{R}_{\geq 0}$. Then, α-PEPP is a class of problems reducible to α-Empty: given $C\colon [M] \to [N]$ find $y \in [N]$ not in the image of C, where $N/M > 1 + \alpha$.

Theorem ([KKMP21])

For any $\alpha(n) > 1/\mathsf{poly}(n)$, α-PEPP $\subseteq \mathsf{FP}^{\mathsf{NP}}_{\mathsf{poly}}$.

Proof

Consider a circuit C mapping $[M]$ to $[N]$, where $N/M > 1 + \alpha$. Let $R(C)$ denote the range of C.
The probability that a random $y \in [N]$ belongs to $R(C)$ is at most $M/N < 1/(1+\alpha)$. Hence, if $k = \lceil n/\alpha \rceil$ there there are $y_1, y_2, \ldots, y_k$ containing a valid solution to *every* length-n instance of α-Empty.

Abundant Avoidance (II)

Denote $\mathsf{APEPP} = 1\text{-}\mathsf{PEPP}$ and $\textsc{Avoid}= 1\text{-}\textsc{Empty}$.

Theorem ([KKMP21])

If $1/\mathsf{poly}(n) \leq \alpha \leq 2^{\mathsf{poly}(n)}$, *then* α-PEPP *and* APEPP *are equivalent under* $\mathsf{FP}^{\mathsf{NP}}$ *reductions.*

Proof

We can reduce β-Empty to γ-Empty as follows.

Brain Teaser: Any ideas how to increase the gap?

Abundant Avoidance (II)

Denote $\mathsf{APEPP} = 1\text{-}\mathsf{PEPP}$ and $\textsc{Avoid}= 1\text{-}\textsc{Empty}$.

Theorem ([KKMP21])

If $1/\mathsf{poly}(n) \le \alpha \le 2^{\mathsf{poly}(n)}$, *then* α-PEPP *and* APEPP *are equivalent under* $\mathsf{FP}^{\mathsf{NP}}$ *reductions.*

Proof

We can reduce β-Empty to γ-Empty as follows. Given a circuit that computes a function $C : [M] \to [N]$, let $k = k(n)$, construct a circuit $C' : [M^k] \to [N^k]$:

$$[M^k] \to [M]^k \xrightarrow{C^k} [N]^k \to [N^k].$$

Assuming $N/M > 1 + \beta(n)$, we have $N^k/M^k > (1 + \beta(n))^k \ge 1 + \gamma(n)$, for

$$k(n) \coloneqq \left\lceil \log_{1+\beta(n)}(1 + \gamma(n)) \right\rceil \le \mathrm{poly}(n).$$

Hence, by solving an instance of γ-Empty, we obtain a k-tuple $(y_1, \ldots, y_k)$ not in the image of C^k. Then we need to find y not in the range of C.

Content

ε-Hard

Definition

ε-Hard: given 1^N, output $x \in \{0,1\}^N$ such that $\text{size}(x) \geq N^\varepsilon$.

Theorem ([Kor21])

AVOID *reduces to* ε-Hard *under* P^{NP} *reductions for any* $\varepsilon \in (0,1)$.

Let C be an instance of 2^n-EMPTY, and let $k = 2\lceil \log |C| \rceil \lceil \frac{1}{\varepsilon} \rceil$. Consider the following map $C^* : \{0,1\}^n \to \{0,1\}^{2^k n}$, defined informally as follows: given a string $x \in \{0,1\}^n$, apply C once to get 2 n-bit strings, then apply C to both of those n-bit strings to get four, and continue k times until we have 2^k n-bit strings.

ε-Hard (II)

We will show the following:

1. Let $m = n2^k = \text{poly}(|C|)$, any solution to ε-Hard on input 1^m will be a string that is not in the range of C^*.
2. Given a string outside the range of C^*, we can find a string outside the range of C.

First, any string in the range of C^*, when interpreted as a truth table of length $m = n2^k$ on $\lceil \log n \rceil + k$ variables, can be computed by a circuit of size $O(|C|k)$.

Brain Teaser: Why?

ε-Hard (II)

We will show the following:

1. Let $m = n2^k = \text{poly}(|C|)$, any solution to ε-Hard on input 1^m will be a string that is not in the range of C^*.
2. Given a string outside the range of C^*, we can find a string outside the range of C.

First, any string in the range of C^*, when interpreted as a truth table of length $m = n2^k$ on $\lceil \log n \rceil + k$ variables, can be computed by a circuit of size $O(|C|k)$.
Thus, we now know that any solution to ε-HARD on input 1^m will not be in the range of C^*. Let y be any string outside the range of C^*.

Brain Teaser: How to find a string outside the range of C in P^{NP}?

Consequences

Theorem

There exists a language in E^{NP} *with circuit complexity* $2^{\Omega(n)}$ *if and only if there is a* P^{NP} *algorithm for* Avoid.

- Let $L \in \mathsf{E}^{\mathsf{NP}}$ with $\text{size}(L) \geq 2^{\varepsilon n}$
- Then we have a polynomial-time algorithm for ε'-Hard.

Brain Teaser: How?

Consequences

Theorem

There exists a language in E^{NP} *with circuit complexity* $2^{\Omega(n)}$ *if and only if there is a* P^{NP} *algorithm for* Avoid.

- Let $L \in \mathsf{E}^{\mathsf{NP}}$ with $\text{size}(L) \geq 2^{\varepsilon n}$
- Then we have a polynomial-time algorithm for ε'-Hard.
- Alternatively, say there is a P^{NP} algorithm for Avoid.
- Then, there is a P^{NP} algorithm for ε-Hard for any fixed $\varepsilon < \frac{1}{2}$.

Brain Teaser: Then what?

Consequences

Theorem

There exists a language in E^{NP} *with circuit complexity* $2^{\Omega(n)}$ *if and only if there is a* P^{NP} *algorithm for* Avoid.

- Let $L \in \mathsf{E}^{\mathsf{NP}}$ with $\text{size}(L) \geq 2^{\varepsilon n}$
- Then we have a polynomial-time algorithm for ε'-Hard.
- Alternatively, say there is a P^{NP} algorithm for Avoid.
- Then, there is a P^{NP} algorithm for ε-Hard for any fixed $\varepsilon < \frac{1}{2}$.
- Consider the language L decided by the following $\mathsf{E}^{\Sigma_{i+1}^P}$ machine: given an n-bit input, on input 1^{2^n}, generate a hard truth table, then look up the n-bit input in this truth table to determine whether to accept or reject.
- By definition this language must have circuit complexity $2^{\Omega(n)}$, and this machine will run in time $\text{poly}(2^n) = 2^{O(n)}$ with an NP oracle.

Consequences (II)

Theorem ([Kor21])

E^{NP} (resp. $\mathsf{EXP}^{\mathsf{NP}}$) contains a language of circuit complexity $2^{\Omega(n)}$ (resp. $2^{n^{\Omega(1)}}$) if and only if E^{NP} (resp. $\mathsf{EXP}^{\mathsf{NP}}$) contains a language of circuit complexity $\frac{2^n}{2n}$.

- If there is a language in E^{NP} of circuit complexity $2^{\Omega(n)}$, then there is a P^{NP} algorithm for Avoid
- Then, there is a P^{NP} algorithm for HardTruthTable, which finds a language with circuit complexity $\frac{2^n}{2n}$.

Content

Takeaways

- TFNP captures total search; mirrors $\mathsf{NP} \cap \mathsf{coNP}$ at decision level.
- Range-avoidance classes ($\mathsf{PEPP}/\mathsf{APEPP}$) generalize union-bound existence arguments.
- Hard Truth Table is APEPP-complete under P^{NP}; many explicit constructions land here.
- Circuit complexity amplification for E^{NP} and $\mathsf{EXP}^{\mathsf{NP}}$.

Bibliography I

[Meg88] Nimrod Megiddo. *A note on the complexity of P-matrix LCP and computing an equilibrium*. IBM Thomas J. Watson Research Division San Jose, CA, 1988.

[MP91] Nimrod Megiddo and Christos H. Papadimitriou. "On Total Functions, Existence Theorems and Computational Complexity". In: *Theor. Comput. Sci.* 81.2 (1991), pp. 317–324.

[WY92] Gerhard J. Woeginger and Zhongliang Yu. "On the Equal-Subset-Sum Problem". In: *Inf. Process. Lett.* 42.6 (1992), pp. 299–302.

[BG94] Mihir Bellare and Shafi Goldwasser. "The Complexity of Decision Versus Search". In: *SIAM J. Comput.* 23.1 (1994), pp. 97–119.

Bibliography II

[Pap94] Christos H. Papadimitriou. “On the Complexity of the Parity Argument and Other Inefficient Proofs of Existence”. In: *J. Comput. Syst. Sci.* 48.3 (1994), pp. 498–532.

[Val77] Leslie G. Valiant. “Graph-Theoretic Arguments in Low-Level Complexity”. In: *MFCS*. Vol. 53. Lecture Notes in Computer Science. Springer, 1977, pp. 162–176.

[Pud15] Pavel Pudlák. “On the complexity of finding falsifying assignments for Herbrand disjunctions”. In: *Arch. Math. Log.* 54.7-8 (2015), pp. 769–783.

[Jer16] Emil Jerábek. “Integer factoring and modular square roots”. In: *J. Comput. Syst. Sci.* 82.2 (2016), pp. 380–394.

Bibliography III

[KKMP21] Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos H. Papadimitriou. "Total Functions in the Polynomial Hierarchy". In: *ITCS*. Vol. 185. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 44:1–44:18.

[Kor21] Oliver Korten. "The Hardest Explicit Construction". In: *FOCS*. IEEE, 2021, pp. 433–444.

[LPR24] Yuhao Li, William Pires, and Robert Robere. "Intersection Classes in TFNP and Proof Complexity". In: *ITCS*. Vol. 287. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, 74:1–74:22.