Randomized separations in black-box TFNP

Kiselev Fedor
MIPT, Russia

August 3, 2026

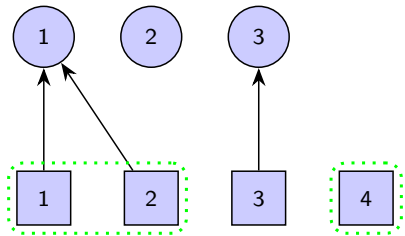
Black-box TFNP

Definition

A *search problem* is a sequence $R = \{R_N\}_{N \in \mathbb{N}}$, where $R_N \subset \Sigma_N^{l_N} \times O_N$ for some finite *alphabet* Σ_N, a finite *set of solutions* O_N, and *input length* l_N. Any $x \in \Sigma_N^{l_N}$ is called the *input* of R_N. A solution o is *correct* for an input x if $(x,o) \in R_N$. A search problem is considered *total* if, for any input, there is a correct solution. Search problem R belongs to TFNP if it is total and there is a family of decision trees $T = \{T_{N,o}\}_{N \in \mathbb{N}, o \in O_N}$ of depth $\mathrm{poly}(\log N)$ such that $(x,o) \in R_N \Leftrightarrow T_{N,o}(x) = 1$.

TFNP subclasses: PPP

Search problem PIGEON: Given $f : [N] \to [N-1] \cup \{\perp\}$, find $i_1 \neq i_2$ such that $f(i_1) = f(i_2)$ or i such that $f(i) = \perp$.

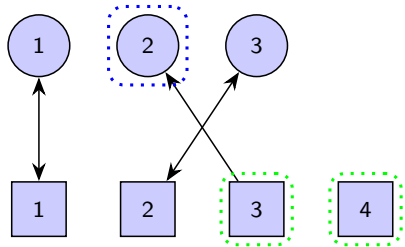


The set of TFNP problems that can be reduced to PIGEON defines TFNP subclass, called PPP.

TFNP subclasses: PPAD and PPADS

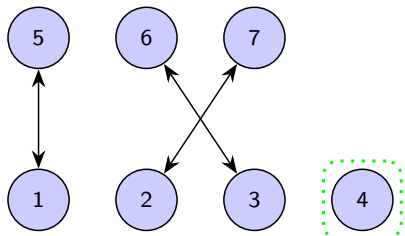
InjPigeon: Given $f : [N] \to [N-1] \cup \{\perp\}$ and $g : [N-1] \to [N] \cup \{\perp\}$, find i such that $f(i) = \perp$ or $g(f(i)) \neq i$. Defines TFNP subclass PPADS.

BijPigeon is defined similarly, but we also accept as a solution j such that $g(j) = \perp$ or $f(g(j)) \neq j$. Defines TFNP subclass PPAD.



TFNP subclasses: PPA

LONELY: Given $f : [2N+1] \to [2N+1] \cup \{\perp\}$, find i such that $f(i) \in \{\perp, i\}$ or $f(f(i)) \neq i$. Defines TFNP subclass PPA.

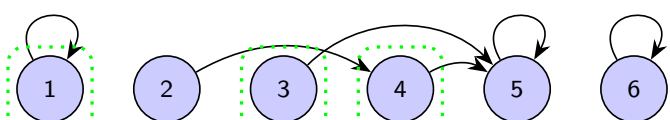


TFNP subclasses: PLS

SoD (Sink of DAG): Given $f : [N] \to [N]$ such that $f(i) \geq i$, we accept as a solution:

- 1 if $f(1) = 1$;
- i if $f(i) > i$, $f(f(i)) = f(i)$;

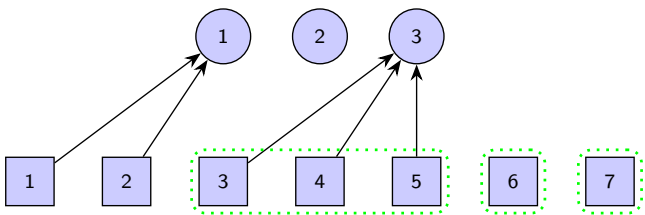
Defines TFNP subclass PLS.



TFNP subclasses: t-PPP

For any nondecreasing $t : \mathbb{N} \to \mathbb{N}$ such that $2 \leq t(N) \leq \mathrm{poly}(\log N)$ we can define a search problem t-PIGEON: Given $f : [(t-1)N + 1] \to [N] \cup \{\perp\}$, find either t-collision or i such that $f(i) = \perp$. Defines TFNP subclass t-PPP.

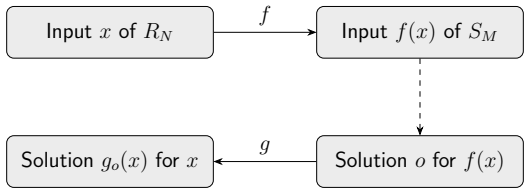
Example for $t = 3$:



Reductions between TFNP problems

Reduction from R_N to S_M consists of:

- $f = \{f_i\}$: family of decision trees over input of R_N, one for each position in input of S_M.
- $g = \{g_o\}$: family of decision trees over input of R_N, one for each possible solution of S_M.



Reduction successful on x if $\forall o : [(f(x), o) \in S_M \Rightarrow (x, g_o(x)) \in R_N]$.

Reductions between TFNP problems

Complexity of a reduction: maximum decision tree depth $+ \log M$.

Reduction from R to S is a sequence of reductions from R_N to $S_{M(N)}$, which are successful on all inputs and have $\mathrm{poly}(\log N)$ complexity.

Reduction from R to S is a sequence of distributions on reductions from R_N to $S_{M(N)}$, which have $\mathrm{poly}(\log N)$ complexity and a success rate of at least $\frac{1}{\mathrm{poly}(\log N)}$ for all inputs.

Reductions between TFNP problems

Using Yao's minimax principle, we can rewrite randomized reducibility as follows:

Lemma

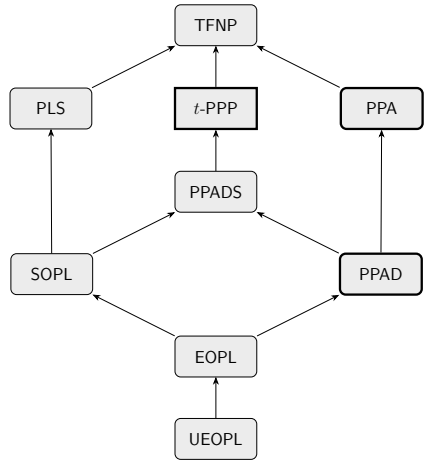
R is randomized reducible to S iff for some $M = M(N)$ for any N and distribution $\mathcal{X}$ of inputs R_N there is a (deterministic) reduction from R_N to S_M with complexity $\mathrm{poly}(\log N)$ and success rate at least $\frac{1}{\mathrm{poly}(\log N)}$ for $x \sim \mathcal{X}$.

Class diagram in black-box model

$R \longrightarrow S$: R is reducible to S.

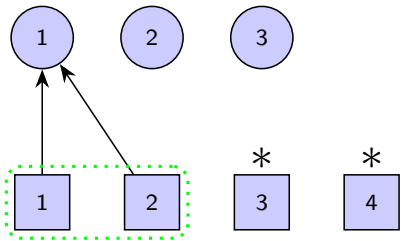
It is known that this diagram is complete, i.e. there is **no** other reductions between classes.

For highlighted classes we have shown that any randomized reduction from them can be derandomized.

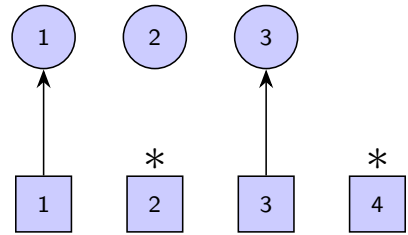


Partial assignment

Partial assignment represents information we managed to learn about a given input. We say that partial assignment is *witnessing* if it is sufficient to give a correct solution.



Witnessing



Non-witnessing

Shown above partial assignments are *consistent*, but they do not *extend* each other.

Honest and perceptive reductions

Properties of a reduction (f,g) from R_N to S_M:

- *Honest*: For any x and g_o, either $g_o(x) = \perp$ or computation of g_o on x witnesses solution $g_o(x)$.
- *Perceptive*:
  1. All g_o start by running $T^S_{M,o} \circ f$, i.e. by checking that $(f(x), o) \in S_M$.
  2. If computation of g_o on x witnesses any solution, then $g_o(x)$ is one of them.

Both properties are easily achievable.

Honest and perceptive reductions

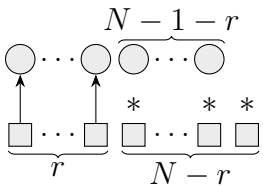
Lemma

Let (f,g) be an honest and perceptive reduction of R_N to S_M of depth h. Then, for any bad input x, there is a partial assignment ρ such that:

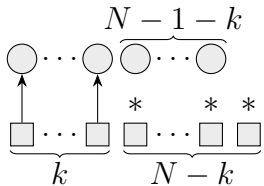
- *ρ is non-witnessing,*
- *$|\rho| \leq h$,*
- *$\rho \sqsubset x$,*
- *Any x' such that $\rho \sqsubset x'$ is also bad.*

Proof idea: we can take as ρ computation of g_o on x for some bad o.

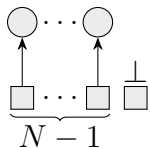
Useful sets



(a) The structure of τ.



(b) The structure of κ.



(c) The structure of $x \in X_N$.

Here $r = \mathrm{poly}(\log N)$, $k = N - N^{\varepsilon}$.

About X_N

Lemma

If PPP *is reducible to* S *on inputs* $x \in X_N$*, then* PPP *is reducible to* S *on all inputs.*

Idea:

1. We can make reduction honest and perceptive.
2. Any non-witnessing partial assignment can be extended to $x \in X_N$.

Main result outline

Our goal: assuming that PPP is not reducible to S, show that all low-complexity reductions from PPP to S perform bad for x taken uniformly from X_N. Plan:

1. Make reduction honest and perceptive.

Main result outline

Our goal: assuming that PPP is not reducible to S, show that all low-complexity reductions from PPP to S perform bad for x taken uniformly from X_N. Plan:

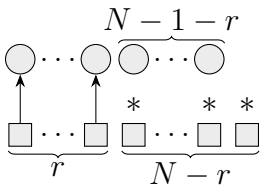
1. Make reduction honest and perceptive.
2. For some set T: input x is bad iff it extends some $\tau \in T$.

Main result outline

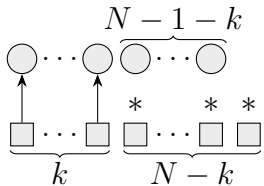
Our goal: assuming that PPP is not reducible to S, show that all low-complexity reductions from PPP to S perform bad for x taken uniformly from X_N. Plan:

1. Make reduction honest and perceptive.
2. For some set T: input x is bad iff it extends some $\tau \in T$.
3. Every κ consistent relative to X_N with some $\tau \in T$.

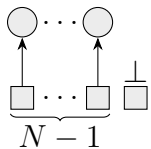
Useful sets



(a) The structure of τ.



(b) The structure of κ.



(c) The structure of $x \in X_N$.

Here $r = \mathrm{poly}(\log N)$, $k = N - N^{\varepsilon}$.

What if there is κ, inconsistent with all $\tau \in T$?

Idea: we can restrict reduction from PPP_N to S_M by κ to get reduction from $\mathrm{PPP}_{N^\varepsilon}$ to S_M. Moreover, the resulting reduction will be always successful on X_{N^ε}.

Lemma

Given a reduction (f,g) from (R_N, X_N) to a search problem S_M and a partial assignment κ, if $x' \in X_N \restriction \kappa$ is a bad input for reduction $(f,g)\restriction\kappa$, then $x = \kappa \cup x'$ is bad for reduction (f,g).

Main result outline

Our goal: assuming that PPP is not reducible to S, show that all low-complexity reductions from PPP to S perform bad for x taken uniformly from X_N. Plan:

1. Make reduction honest and perceptive.
2. For some set T: input x is bad iff it extends some $\tau \in T$.
3. Every κ consistent relative to X_N with some $\tau \in T$.
4. For any τ: $|\{\kappa : \kappa \sqsupset \tau\}| \geq (1 - o(N^{-\delta}))|\{\kappa : \kappa \overset{X_N}{\Uparrow} \tau\}|$.

Main result outline

Our goal: assuming that PPP is not reducible to S, show that all low-complexity reductions from PPP to S perform bad for x taken uniformly from X_N. Plan:

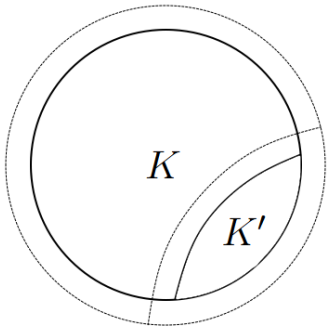
1. Make reduction honest and perceptive.
2. For some set T: input x is bad iff it extends some $\tau \in T$.
3. Every κ consistent relative to X_N with some $\tau \in T$.
4. For any τ: $|\{\kappa : \kappa \sqsupset \tau\}| \geq (1 - o(N^{-\delta}))|\{\kappa : \kappa \overset{X_N}{\Uparrow} \tau\}|$.
5. $|\{\kappa : \exists \tau \in T, \kappa \sqsupset \tau\}| \geq (1 - o(N^{-\delta}))|\{\kappa : \exists \tau \in T, \kappa \overset{X_N}{\Uparrow} \tau\}|$.

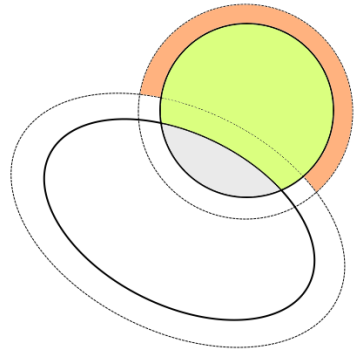
Consistency to extension ratio

Main ideas:

1. $\kappa \overset{X_N}{\uparrow\uparrow} \tau$ iff $\kappa \overset{X_N}{\uparrow\uparrow} \kappa_2$ for some $\kappa_2 \sqsupset \tau$.
2. Fix arbitrary τ. Let $K = \{\kappa \mid \kappa \sqsupset \tau\}$. Then for any $K' \subset K$ we have $|\{\kappa \mid \exists \kappa_2 \in K' : \kappa \overset{X_N}{\uparrow\uparrow} \kappa_2\}|/|K'| \geq |\{\kappa \mid \exists \kappa_2 \in K : \kappa \overset{X_N}{\uparrow\uparrow} \kappa_2\}|/|K|$
3. Induction by the size of T.

Consistency to extension ratio





Main result outline

Our goal: assuming that PPP is not reducible to S, show that all low-complexity reductions from PPP to S perform bad for x taken uniformly from X_N. Plan:

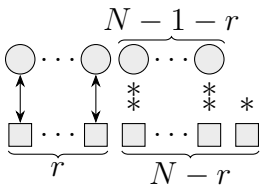
1. Make reduction honest and perceptive.
2. For some set T: input x is bad iff it extends some $\tau \in T$.
3. Every κ consistent relative to X_N with some $\tau \in T$.
4. For any τ: $|\{\kappa : \kappa \sqsupset \tau\}| \geq (1 - o(N^{-\delta}))|\{\kappa : \kappa \overset{X_N}{\Uparrow} \tau\}|$.
5. $|\{\kappa : \exists \tau \in T, \kappa \sqsupset \tau\}| \geq (1 - o(N^{-\delta}))|\{\kappa : \exists \tau \in T, \kappa \overset{X_N}{\Uparrow} \tau\}|$.
6. Almost all κ extend some $\tau \in T$.

Main result outline

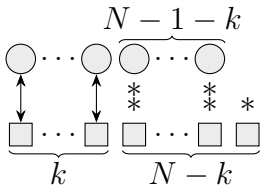
Our goal: assuming that PPP is not reducible to S, show that all low-complexity reductions from PPP to S perform bad for x taken uniformly from X_N. Plan:

1. Make reduction honest and perceptive.
2. For some set T: input x is bad iff it extends some $\tau \in T$.
3. Every κ consistent relative to X_N with some $\tau \in T$.
4. For any τ: $|\{\kappa : \kappa \sqsupset \tau\}| \geq (1 - o(N^{-\delta}))|\{\kappa : \kappa \overset{X_N}{\Uparrow} \tau\}|$.
5. $|\{\kappa : \exists \tau \in T, \kappa \sqsupset \tau\}| \geq (1 - o(N^{-\delta}))|\{\kappa : \exists \tau \in T, \kappa \overset{X_N}{\Uparrow} \tau\}|$.
6. Almost all κ extend some $\tau \in T$.
7. Almost all inputs are bad.

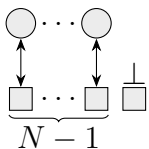
Same for PPAD



(a) The structure of τ.

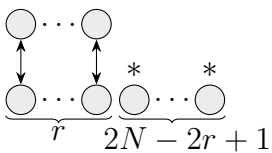


(b) The structure of κ.

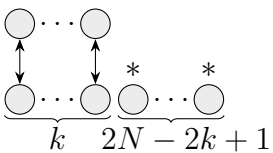


(c) The structure of $x \in X_N$.

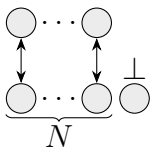
Same for PPA



(a) The structure of τ.

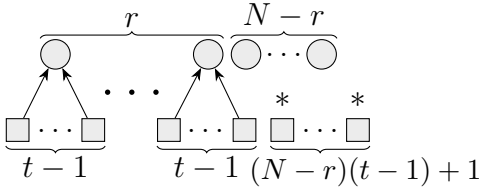


(b) The structure of κ.

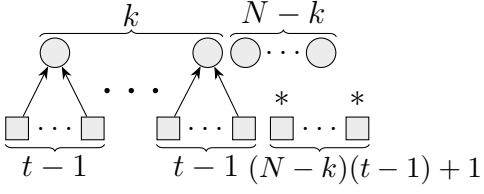


(c) The structure of $x \in X_N$.

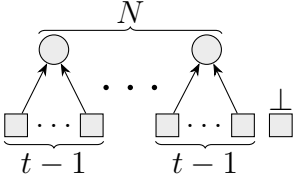
Same for t-PPP



(a) The structure of τ.



(b) The structure of κ.



(c) The structure of $x \in X_N$.

Appendix

Search problem (black-box)

Definition

A *search problem* is a sequence $R = \{R_N\}_{N \in \mathbb{N}}$, where $R_N \subset \Sigma_N^{l_N} \times O_N$ for some finite *alphabet* Σ_N, finite *set of solutions* O_N and *input length* $l_N \geq N$. Any $x \in \Sigma_N^{l_N}$ is called *input* of R. $o \in O_N$ called *correct solution* for input x if $(x,o) \in R_N$. A search problem is *total* if for any input there is a correct solution. R belongs to TFNP^{dt} if it is total and there is a family of decision trees $T = \{T_{N,o}\}_{N \in \mathbb{N}, o \in O_N}$ of depth $\mathrm{poly}(\log N)$ such that $\forall N\ \forall x \in \Sigma_N^{l_N}\ \forall o \in O_N : (x,o) \in R_N \Leftrightarrow T_{N,o}(x) = 1$.

Reduction (black-box)

Definition

Given search problems R_N and S_M, a *reduction* from R_N to S_M is a pair $(f = \{f_i\}_{i\in[l_M]}, g = \{g_o\}_{o\in O_M})$ of families of decision trees over inputs of R_N. We say that the reduction is successful on input x if:

$$\forall o \in O_M : [(x, g_o(x)) \in R_N \Leftarrow (f(x), o) \in S_M],$$

where $f(x) = f_1(x)f_2(x)\dots f_{l_M^{(S)}}(x)$. *Depth* of reduction d is the maximum depth of decision trees in f and g. *Complexity* of reduction is $d + \log M$. R is *reducible* to S if for every N there is $M = M(N)$ and a reduction of R_N to S_M which is successful on all inputs of R_N and has complexity $\mathrm{poly}(\log N)$.

Randomized reduction (black-box)

Definition

Given search problems R_N and S_M, a *randomized reduction* from R_N to S_M is a distribution D over some finite set of reductions from R_N to S_M. *Rate of success* on input x is:

$$\mathbb{P}_{(f,g)\sim D}[\forall o \in O_M : (x, g_o(x)) \in R_N \Leftarrow (f(x), o) \in S_M].$$

Depth of randomized reduction equals the maximum of depths of reductions in D. Complexity is defined the same way. R is *randomized reducible* to S if for every N there is $M = M(N)$ and a randomized reduction of R_N to S_M with the rate of success at least $\frac{1}{\mathrm{poly}(\log N)}$ on all inputs and complexity $\mathrm{poly}(\log N)$.