TFNP

FNP: relation analogue of NP

<u>Def</u> a relation R is in FNP if $R(x,y)$ can be determined in polynomial time and $|y| = O(\text{poly}\,|x|)$ for all $(x,y) \in R$

TFNP: subset of FNP consisting of those relations which are <u>total</u>

<u>Def</u> a relation R is in TFNP if $\forall x, \exists y$ s.t. $|y| = O(\text{poly}|x|)$ and $(x,y) \in R$, and membership in R can be determined in polynomial time

we call y a "witness" or "solution"

TFNP relations/problems are often based on non-constructive existence theorems

- pigeonhole principle
- fixed point theorems
- combinatorial graph arguments (ex: Ramsey's Theorem)

EXAMPLE: PIGEONHOLE PROBLEM

Suppose $f: \{0,1\}^n \to \{0,1\}^n$

then either:

- $\exists\ x$ s.t. $f(x) = 0^n$

  or

- $\exists\ x_1, x_2$ s.t. $x_1 \neq x_2$ and $f(x_1) = f(x_2)$

Q: Can you think of a TFNP relation inspired by this principle?

PIGEONHOLE PROBLEM

R contains tuples of the form

(f, x) where $f(x) = 0^n$

$(f, (x_1, x_2))$ where $x_1 \neq x_2$ and $f(x_1) = f(x_2)$

Def PPP is the class of problems reducible to PIGEONHOLE

PPP has lots of interesting connections!

<u>Theorem</u> [Papadimitriou '94]

If PPP = FP then one-way permutations do not exist.

↳ like FNP, but finding y given x must be poly time

proof idea:

given x and algo for computing π

turn algo into a circuit and XOR output with x

resulting circuit C has no collisions and $C(x) = 0^{|x|}$

PLS: polynomial local search

find locally optimal solution

ex: MAX-CUT

given graph $G=(V,E)$

solution is partition V_1, V_2 of the vertices s.t. the # of edges between V_1, V_2 cannot be increased by moving one vertex from one set to the other

MORE PLS:

- MAX-2SAT
- finding Nash equilibria (in certain settings)

fun fact: PLS was defined by Johnson, Papadimitriou, Yannakakis in 1988

Many TFNP problems concern functions

- PIGEONHOLE
- local MAX-CUT
  - $f(V_1, V_2) :=$ # edges btwn V_1 and V_2
- finding a fixed point in a monotone function

Q: how much harder do such problems become when algo has only black box access to the function in question?

EXAMPLE: PIGEONHOLE in the black box

Recall: given input $f: \{0,1\}^n \to \{0,1\}^n$
want to find either:

- $x_1 \neq x_2$ s.t. $f(x_1) = f(x_2)$
- x s.t. $f(x) = 0^n$

Q: is this hard (in n) in the black box setting?

First, definitions:

Def [Komargodski, Naor, Yogev] (subset of TFNP that makes sense in black box setting)

a TFNP problem is a relation R where for every $f : \{0,1\}^n \to \{0,1\}^n$, $\exists\ x_1, \ldots, x_{q(n)} \in \{0,1\}^n$ s.t.
$(x_1, \ldots, x_{q(n)}, f(x_1), \ldots, f(x_{q(n)})) \in R$
where $q(\cdot)$ is a polynomial

NOTE:

not all TFNP problems make sense in the black box setting

ex: R consisting of pairs (x, x)

Theorem [KNY] given a TFNP relation R and black box access to a function f, we can always find a solution $(x_1, \ldots, x_{q(n)}, f(x_1), \ldots, f(x_{q(n)}))$ with $O(\mathrm{poly}|f|)$ queries to the black box.

Q: why doesn't this contradict our observation of PIGEONHOLE being black box hard?

PROOF.

Algorithm. suppose $|f|$ is known

- initialize list L of all fns of length $\leq |f|$
- while $|L| > 0$:
  - define f^* s.t. $f^*(x) := \text{MostFrequent}_L(x)$
  - find a solution s for f^*; query its points
  - if BB agrees, output s

    else, remove all fns disagreeing with queries from L

ANALYSIS.

- in each iteration, $|L|$ is (at least) halved
- need at most $\log |L| = |f|$ iterations
- algo makes $O(\text{poly}|f|)$ queries per iteration

Q: What if $|f|$ is not known?

LIMITATIONS

1. Solution size assumed $O(\text{poly}(n))$

   $\hookrightarrow$ len of points in domain of $f: \{0,1\}^n \to \{0,1\}^n$

   - general TFNP defn:

     $(x,y) \in R \qquad |y| = O(\text{poly}|x|)$

     $\hookrightarrow$ len of representation of f

2. Solution must be verifiable with $O(\text{poly}|f|)$ queries to the BB

   - does not work for $R = \{(f_1, f_1), (f_2, f_2), \ldots\}$

3. algo seems very inefficient $(|L| = 2^{|f|})$

CAN WE IMPROVE ALGO? (ongoing work w/ Mihalis)

1. Solution size assumed $O(\text{poly}(n))$
   → can remove this assumption for "truly" total problems
2. Solution must be verifiable with $O(\text{poly}|f|)$ queries to the BB
   → no hope to remove this assumption...
3. algo seems very inefficient
   → actually in poly. hierarchy