coNP and Function Problems

Petros Barbagiannis
$\mu\Pi\lambda\forall$

February 17, 2014

coNP

Definition: $\mathbf{coNP} = \{L : \bar{L} \in \mathbf{NP}\}$

For every $L \subseteq \{0,1\}^*$, $L \in \mathbf{coNP}$ if there exists a polynomial $p: \mathbb{N} \to \mathbb{N}$ and a polynomial time TM M s.t. for every $x \in \{0,1\}^*$,

$$x \in L \Leftrightarrow \forall u \in \{0,1\}^{p(|x|)}, M(x,u) = 1$$

coNP

If a language L is in **coNP** and every other language in **coNP** is polynomial-time Karp reducible to L, then L is **coNP**-complete.

TAUTOLOGY $= \{\varphi : every\ truth\ assignment\ satisfies\ \varphi\}$

<u>TAUTOLOGY is **coNP**-complete:</u>

Let $L \in$ **coNP**. Then $\bar{L} \in$ **NP**. By Cook-Levin theorem, there is a reduction R from $\bar{L}$ to SAT. The reduction from L to TAUTOLOGY is $R' = \neg R$.

$\mathbf{NP} \cap \mathbf{coNP}$

$L \in \mathbf{NP} \cap \mathbf{coNP}$ if $\forall x \in \{0,1\}^*$,

$$x \in L \Rightarrow \exists u \in \{0,1\}^{p(|x|)} \; s.t. \, M(x,u) = 1$$
$$x \notin L \Rightarrow \exists u \in \{0,1\}^{p(|x|)} \; s.t. \, M(x,u) = 0$$

FACTORING (D): Given two integers N, k does N have a prime factor $d < k$?

FACTORING $\in \mathbf{NP} \cap \mathbf{coNP}$

*When two optimization problems are dual to each other, the decision versions of these problems are both in $\mathbf{NP} \cap \mathbf{coNP}$.

Function Problems

Definition: **FNP**

Let $L \in \mathbf{NP}$. There is a polynomially decidable relation R_L that for every input x there is a string y, $|y| \leq |x|^{O(1)}$ s.t. $R_L(x, y) \Leftrightarrow x \in L$.
A function problem FL associated with L is the following:
given x, find any y s.t. $R_L(x, y)$. If no such y exists then return "no“.

FP: The subclass of **FNP** problems that can be solved in polynomial time.

$$\mathbf{FP} \subseteq \mathbf{TFNP} \subseteq \mathbf{FNP}$$
$$\mathbf{FP} = \mathbf{FNP} \Leftrightarrow \mathbf{P} = \mathbf{NP}$$

Function Problems

FSAT: Given a Boolean expression φ, if φ is satisfiable return any satisfying truth assignment, otherwise return "no".

FSAT $\in$ **FNP**

If SAT can be solved in polynomial time, so can FSAT. Let A_{SAT} be a polynomial time algorithm for SAT. Given an expression φ with variables $x_1, \ldots, x_n$ first check whether φ is satisfiable. If A_{SAT} says "no", then return "no". If it is satisfiable, by making $2n$ calls of A_{SAT}, each time assigning true and false to variable x_i, and substituting the value that satisfies φ we can find a satisfying truth assignment for φ.

Total Function Problems

Definition: TFNP

$L \in$ **TFNP** if for every string x there is at least one y such that $R_L(x, y)$, that is, the function computing y is total.

FACTORING $\in$ **TFNP**

TFNP Reductions

TFNP reduction from problem L to problem K is the following two functions:

- a function f that maps each instance x from L to an instance $f(x)$ of K
- a function g that, for each instance x of L and answer y for instance $f(x)$ yields an answer $g(x, y)$ for x

PLS

Definition: **PLS**

$L \in$ **PLS**

- y (solution) is polynomially bounded in the size of x (input)
- Polynomial time algorithm that $\forall x$ determines whether x is an instance of L and if so, outputs an initial solution for L
- Polynomial time algorithm that given x, y, determines whether y is a solution for x and if so, outputs an integer value $c(x, y)$
- Polynomial time algorithm that given x, y, either reports "locally optimal" or moves to a new neighbor/solution

PLS Reductions

Let $L, K \in$ **PLS**

L *is* **PLS**-reducible to K if there exist polynomial time computable functions f and g s.t.:

- f maps instances x of L to instances $f(x)$ of K
- g maps $(solution\ of\ f(x), x)$ to solutions of x
- $\forall x$ of L if s is a local optimum for instance $f(x)$ of K, then $g(s, x)$ is local optimum for x

L is **PLS**-complete if every problem in **PLS** is **PLS**-reducible to L.

PLS Reductions

CIRCUIT FLIP:

Given a feedback-free Boolean circuit composed of AND, OR and NOT gates, with m inputs and n outputs, find an input string such that the output cannot be improved lexicographically by flipping a single input bit. A solution is any m-bit vector s.
A neighbor of s is any vector that can be obtained from s by changing exactly one bit.

CIRCUIT FLIP is **PLS**-complete

Proof: see [3].

PPAD

$L \in \mathbf{PPAD_0}$ if:

- y is polynomially bounded
- a polynomial time algorithm that given a string x determines whether x is an instance of L and if so outputs an initial source solution y_0
- a polynomial time algorithm that given an instance x and a string y determines whether y is a solution for x and if so, outputs a string $pred(x, y) = y'$
- a polynomial time algorithm that given an instance x of L and a string y determines whether y is a solution for x and if so outputs a string $succ(x, y) = y'$

A problem is in **PPAD** if it is **TFNP-**reducible to a $\mathbf{PPAD_0}$ −complete problem.

END OF THE LINE $\in \mathbf{PPAD_0}$

Bibliography

1. C. H. Papadimitriou, ***Computational Complexity***, 1993
2. S. Arora, B. Barak, ***Computational Complexity-A modern approach,*** 2007
3. D. Johnson, C. Papadimitriou, M. Yannakakis, ***How easy is local search?***, 1988
4. D. Johnson, ***The NP-completeness Column: Finding Needles in Haystacks,*** 2007
5. C. Daskalakis, P. Goldberg, C. Papadimitriou, ***The Complexity of Computing a Nash Equilibrium,*** 2008