Counting Problem of *PureCircuit*

Christos Demetriou *Department of Computer Science*
University of Warwick
Coventry, UK
u2018918@live.warwick.ac.uk, u2018918

***Abstract*—Test.Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.**

***Index Terms*—Complexity Theory, Counting Complexity, Hazard Free Circuits, TFNP, PPAD, Search Problems, Kleene Logic**

I. Introduction

Over the last couple of years, there has been a revolutionary initiative in the field of combinatronics. Combinatronics has been a field of study in mathematics that primarly focused on the notion of counting objects with certain properties. Over time, this notion has shifted, especially in the subfield of algebraic combinatronics, where there is no clear notion of the object that we are counting and the numbers express something more abstract [1]. This gave a need to be able to assign a combinatorial interpretation to such numbers, or more simply, do these numbers correpsond to some counting over a set of objects. Being able to find such definitions or interpretations can be very important, it allows us to utilise tools from combinatronics as well as allow us to understand and reveal hidden structures and properties for such numbers [1]. Moreover there are several problems or numbers such as *Kronecker coefficients*, whose combinatorial interpretation, would give a step towards the resolution of the $P \neq NP$ conjecture [2].

To reiterate the previous statement, we can understand combinatorial interpretation as the process of: given a sequence of numbers $\{a_x\}$, find a set of combinatorial objects A_x such that $|A_x| = a_x$ To formalise the current idea, Igor Pak et al. has concluded that $f \in \#P$, implies that f has a combinatorial interpretation [1], [2]. We will explore this idea in much greater detail in the upcoming section, but the main benefit is the ability to use a very expressive but formal language that encapsulates this notion of a combinatorial interpretation.

In our current work, we focus on extending the work done by Ikenmeyer et al., where they focused on the creation of frameworks that determine whether $f \in^? \mathbf{\#P}$, by looking at the complexity class of #*TFNP -1*. This is a class of problems that are guaranteed to have a solution and their solutions are verifiable in polynomial time. In their paper, they were able to show that for a subclass of problems, also known as *PPAD* $\subseteq$ *TFNP*, different *PPAD-complete* problems, may or may not have a combinatorial interpretation. Our contribution, comes to the analysis of a specific problem, known as PureCircuit, which utilises Kleene logic, to find satisfying assingments in sequential circuits. We hope to demonstrate that such problem could help us bound, the counting complexity limits of #PureCircuit $- 1$.

A. Project objectives

Below we will present our table of objectives. We will denote updated objectives with *, new objectives with (!), completed objectives with (✓), deleted objectives with (-).

R.1) (✓) Find a parsimonious reduction from the *EndOfLine* to *EndOfLine*.
R.2) (!) Improve the combinatorial bound between the reduction from EndOfLine to PureCircuit
R.3) (!) Demonstrate that #PPAD$(PureCircuit) - 1 \not\subseteq$ **#P**
R.4) (*) Prove or disprove the following $\forall n \in \mathbb{N}_{\geq 2}$:

$$\exists c \in \mathbb{N} : \#\textsc{SourceOrExcess}(n, 1) \subseteq^c \#\textsc{PureCircuit}$$

R.5) (*) Prove or disprove the following claim: $\forall L \in$ **PPAD**

$$\exists c \in \mathbb{N} : \#L \subseteq^c \#\textsc{PureCircuit}$$

Below we will be representing the development portion of the project.

S.1) Visualise a pure circuit
S.2) Generate a solution
S.3) Count number of solutions for smaller scales

We will compile the rest of the report, based on our current findings, how we modified our objectives to the current ones as well as general reflections.

B. Core set of results

Below we will present our main list of findings up to this point:

Definition 1.1 (Poly-Function Bounded Parsimonious Reductions): Given two counting problems $A, B : \{0,1\}* \to \mathbb{N}$ and a function $f : 0,1^* \to \mathbb{N}$ such that $f \in n^{O(1)}$, we say that:

$$A \subseteq^f B$$

If for input $w \in \{0,1\}^*$, if a represent the number of solutions for problem A and b the number of solutions for problem B, we have:

$$a \leq b \leq f(|w|) \cdot a$$

Theorem 1.1 (ND-Brouwer to PureCircuit): Given $f(x) \triangleq 20$

$$\#\textsc{Brouwer}^{f_3} \subset^f \#\textsc{PureCircuit}$$

Corollary 1.1.1: For any $d \in \mathbb{N}_{\geq 2}$, we can define $f(\cdot) = \sum_{i=1}^{d-1} \binom{d}{i} 2^i$ such that:

$$\#\textsc{Brouwer}^{d} \subset^f \#\textsc{PureCircuit}$$

Theorem 1.1: For any $d \in \mathbb{N}_{\geq 2}$, we can define $f(\cdot) = 5$ such that:

$$\#\textsc{Brouwer}^{d} \subset^f \#\textsc{PureCircuit}$$

Throughout our search, we stumbled upon various variants of the problem as well as reductions and claims from other subclasses of PPAD or Hazard-Free logic

Proposition 1.2: Weaker variants of PureCircuit can result to parsimonious reductions to the EndOfLine problem as such:

$$\#\textsc{AcyclicBPureCircuit} \subseteq \#\textsc{EndOfLine}$$
$$\#\textsc{PermutationFreeBPureCircuit} \subseteq \#\textsc{EndOfLine}$$

In addition, we were able to show that for different promise problems, we can find reductions to the PureCircuit problem

Proposition 1.3:

$$\#\textsc{PUnsatHazard} \subseteq \#\textsc{PureCircuit}$$

Lastly we introduce the following proposition

Proposition 1.4: Given function $F : \mathbb{T}^n \to \mathbb{T}^n$ where $\mathbb{T} \triangleq \{0, 1, \perp\}$ and F is a **natural** function:

$$\exists x^\star \in \mathbb{T}^n : F(x^\star) = x^\star$$

The idea above is based on the Tarski Fixed point theorem. Using fixed points with the Kleene logic set, has been used in the past by Kozer as seen in his book [3], but we are extending such ideas to any possible functions or gates that use monotone gates. Doing that allows us to make the following observation

Proposition 1.5:

$$\#\textsc{HazardTarski} \subseteq \#\textsc{PureCircuit}$$

II. Preliminaries and Background review

As mentioned in the introduction, the current project works as an interaction of three different fields: 1. counting complexity and combinatorial interpretation, 2. total search problems and PPAD and 3. Kleene logic.

A. Counting Complexity and Combinatorial Interpretations

As we previously mentioned, we say that an object or a structure f has a combinatorial interpretation if $f \in$ **#P**. **#P** is a complexity class created by Valiant [4], to define a formal combinatorial framework in complexity theory.

Definition 2.1 (#P Complexity Class): **#P** is a class of functions $f : \{0,1\}^* \to \mathbb{N}$ such that: there exists a polynomial time non-deterministic Turing machine $N : \{0,1\}^* \to \{0,1\}$ such that

$$\forall w \in \{0,1\}^* : f(w) = \#acc_N(w)$$

Where $\#acc_N(\cdot)$, denotes the number of accepting paths. Equivalently, we may also use the definition of: There exists a polynomial deterministic TM M, and $p : \mathbb{N} \to \mathbb{N}$ such that $p \in n^{O(1)}$, we have:

$$f(w) = \left| \left\{ v \in \{0,1\}^{p(|w|)} \mid M(w,v) = 1 \right\} \right|$$

#P was initially created by [4], to demonstrate, that even if we have a problem $L \in P$, $\#L$ can be computationally hard to compute, by providing an example of computing the permanent of a 01-matrix, with number of perfect matchings.

As we can see, **#P** allows us to define a set of objects, whose cardinality equals $f(w)$. Core reasoning for choosing **#P** to define combinatorial objects is mainly for the following two reasons [5]:

1) By polynomially bounding words, we avoid cases such as: $f(w) = \{1, \ldots, f(w)\}$
2) Current framework allows us to work with $f(\cdot)$, whose direct computation can be computationally hard

The current framework was used in several papers such as [6] and [5] where they were able to use tools from complexity theory to show that many structures do or do not have a combinatorial interpretation. For the purposes of the current project, we are focusing on [6], where they demonstrated how several TFNP problems, change in complexity as we ignore one of its solutions.

B. Total Search Problems and PPAD

When talking about search problems, we are using the following definition:

Definition 2.2 (Search Problems and Total Search Problems): **Search problems** can be defined as relations $R \subseteq \{0,1\}^* \times \{0,1\}^*$, where given $x \in \{0,1\}^*$, we want to find $y \in \{0,1\}^*$ such that xRy.

Total Search problems are search problems such that:

$$\forall x \in \{0,1\}^*, \exists y \in \{0,1\}^* : xRy$$

Using the above, we can define the following complexity clasess

Definition 2.3 (FP, FNP, TFNP): **FP** are *search problems* such that there exists poly-time TM M such that $M(x) = y$ where xRy.

FNP are *search problems* such that there exists poly-time TM $M : \{0,1\}^* \to \{0,1\}$ and a poly function $p : \mathbb{N} \to \mathbb{N}$ such that:

$$\forall x \in \{0,1\}^*, y \in \{0,1\}^{p(|x|)} : xRy \iff M(x,y) = 1$$

Lastly **TFNP** $= \{L \in$ **TFNP** $\mid L$ is total$\}$

Our current work focuses on a specific subclass of **TFNP** problems which is defined as follows:

Definition 2.4 (EndOfLine problem): Given circuits $S, P \in \{0,1\}^n \to \{0,1\}^n$ such that $S, P \in n^{O(1)}$ we define a directed graph $G = (V, E)$, such that $V = \{0,1\}^n$ and E defined as:

$$E = \{(x, y) \in V^2 : S(x) = y \wedge P(y) = x\}$$

We define are source or sinks $\forall v \in V : deg(v) = (0, 1)$ or $deg(v) = (1, 0)$, respectively. We want to output v that is either a source or a sink.

Definition 2.5 (Levin Reductions): Given a pair of search problems R_A, R_B, a pair of computable time functions (f, g) is called a Levin reduction from $R_A \to R_B$

$$S_R \triangleq \{x \mid \exists y : xRy\}$$
$$R(x) \triangleq \{y \mid xRy\}$$
$$\forall x \in S_{R_A}, y_b \in R(f(x)) : (x, g(x, y_b)) \in R_A$$

*Definition 2.6 (**PPAD** complexity class):* **PPAD** is defined as the set of search problems that are levin reducible to the EndOfLine problem

PPAD has been created by Papadimitriou [7] to demonstrate a subset of problems in **NP** that are guaranteed to have a solution but can be very difficult to find. We will define several problems of interest as they will be referenced in later sections.

Definition 2.7 (SourceOrExcess problem): We define as *SourceOrExcess*$(k, 1)$ for $k \in \mathbb{N}_{\geq 2}$ the search problem as such: Given a poly-sized successor circuit $S : \{0,1\}^n$ and a set of predecssor poly-sized circuits $\{P_i\}_{i \in [k]}$, we define the graph $G = (V, E)$ such that, $V = \{0,1\}^n$ and E as:

$$\forall x, y \in V^2 : (x, y) \in E \iff (S(x) = y) \wedge \bigvee_{i \in [k]} P_i(y) = x$$

We ensure that 0^n is as sink, meaning $\deg(0^n) = (0, 1)$. A valid solution is a vertex v such that *in-deg*$(v) \neq$ *out-deg*(v)

Definition 2.8 (Sperner problem):

These problems have found connections with various other problems such as Nash Equilbria, Fixed point theorems or Sperner Lemmas. The current project looks on the counting complexity of such problems as despite two problems being PPAD-complete, Ikenmeyer et al. has showed several examples where their counting difficulty can differ vastly. Examples of such statements can be summarised as:

$$\textbf{\#PPAD}(\textsc{SourceOrSink}) - 1/2 = \textbf{\#P}$$
$$\textbf{\#PPAD}(\textsc{SourceOrExcess}) - 1/2 = \textbf{\#P}$$

C. Kleene Logic and Hazard-Free Circuits

III. Research Methodology

IV. Project Plan

V. Conclusion

References

[1] I. Pak, "What is a combinatorial interpretation?" Sep. 2022.

[2] C. Ikenmeyer and I. Pak, "What is in #P and what is not?" in *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, Oct. 2022, pp. 860–871.

[3] D. Kozen, *Theory of Computation*, ser. Texts in Computer Science. Springer London, 2006.

[4] L. G. Valiant, "The complexity of computing the permanent," *Theoretical Computer Science*, vol. 8, no. 2, pp. 189–201, Jan. 1979.

[5] C. Ikenmeyer, I. Pak, and G. Panova, "Positivity of the Symmetric Group Characters Is as Hard as the Polynomial Time Hierarchy," *International Mathematics Research Notices*, vol. 2024, no. 10, pp. 8442–8458, May 2024.

[6] C. Ikenmeyer and I. Pak, "What is in #P and what is not?" Apr. 2022.

[7] C. H. Papadimitriou, "On the complexity of the parity argument and other inefficient proofs of existence," *Journal of Computer and System Sciences*, vol. 48, no. 3, pp. 498–532, Jun. 1994.