Electronic Colloquium on Computational Complexity, Report No. 4 (2018)

Circuit Complexity of Bounded Planar Cutwidth Graph Matching

Aayush Ojha[1] and Raghunath Tewari[2]

[1]Indian Institute of Technology, Kanpur
Email: aayushoj@cse.iitk.ac.in
[2]Indian Institute of Technology, Kanpur
Email: rtewari@cse.iitk.ac.in

December 16, 2017

Abstract

Recently, perfect matching in bounded planar cutwidth bipartite graphs (BGGM) was shown to be in ACC^0 by Hansen et al. [8]. They also conjectured that the problem is in AC^0.

In this paper, we disprove their conjecture by showing that the problem is not in $\mathsf{AC}^0[p^\alpha]$ for every prime p. Our results show that the previous upper bound is almost tight. Our techniques involve giving a reduction from Parity to BGGM. A further improvement in lower bounds is difficult since we do not have an algebraic characterization for $\mathsf{AC}^0[m]$ where m is not a prime power. Moreover, this will also imply a separation of $\mathsf{AC}^0[m]$ from P. Our results also imply a better lower bound for perfect matching in general bounded planar cutwidth graphs.

1 Introduction

For a graph $G = (V, E)$ a *matching* $M \subseteq E$ is a set of edges in G such that no two edges in M share a common vertex. We say G has a *perfect matching* if there exists a matching that matches every vertex in G. Since every graph is not guaranteed to have a perfect matching, computing a matching of maximum cardinality is a natural generalization of the perfect matching problem. The computational complexity of the matching problem is a well-studied problem particularly in the context of circuit complexity and derandomization.

In 1965, Edmonds showed that computing maximum matching is in P [4]. In 1979, Lovász gave an efficient randomized parallel algorithm for the perfect matching problem by showing that it is in RNC [10]. The construction version of the problem was also shown to be in RNC [9, 11]. It is an important open question whether matching has an efficient deterministic parallel algorithm, that is, whether it is in NC. Attempts to derandomize the above approaches has proved elusive so far. Recently there has been some progress on this problem. Perfect matching was shown to be in quasiNC for bipartite graphs [7] and in a subsequent paper extended to general graphs [14].

ISSN 1433-8092

Stronger results are known for perfect matching in graphs with bounded treewidth and its subclasses. Elberfeld et al. [5] showed that the problem is in L for graphs with bounded treewidth by proving the logspace versions of Bodlaender's and Courcelle's theorem. This gives a tight bound on the complexity of perfect matching in bounded treewidth graphs since it was already known to be L-hard [3]. In a subsequent paper, Elberfeld et al. showed that given a tree decomposition of the input graph as a term representation, perfect matching for bounded treewidth graphs is in uniform NC^1 and for bounded tree-depth graphs is in uniform AC^0 [6]. The upper bound of NC^1 for bounded treewidth graphs is tight since Barrington showed that the problem is hard for NC^1 under projection reductions [1].

In 2014, Hansen et al. used the characterization of Barrington and Thérien [2] and showed that bipartite perfect matching in graphs with bounded planar cutwidth is in ACC^0 [8]. They also gave a lower bound of AC^0 for the same problem. For perfect matching in general bounded planar cutwidth graphs, they gave a lower bound of $\mathrm{AND} \circ \mathrm{OR} \circ \mathrm{XOR} \circ \mathsf{AC}^0$. In their paper, Hansen et al. also conjectured that perfect matching for bipartite bounded planar cutwidth graphs is in AC^0 and for general bounded planar cutwidth graphs is in $\mathsf{AC}^0[2]$.

1.1 Our Result and Proof Outline

We refute both the conjectures in this paper by giving improved lower bounds for perfect matching in bounded planar cutwidth graphs for both bipartite and general graphs. We show that perfect matching for bounded planar cutwidth graph is not in $\mathsf{AC}^0[p^\alpha]$ for every prime p and $\alpha \in \mathbb{N}$.

To show this improved lower bound we first reduce Parity to perfect matching in bipartite bounded planar cutwidth graphs using a family of AC^0 circuits. This is done by constructing certain graph gadgets as defined in Section 3. This reduction and result by Razborov [12] and Smolensky [13] shows the problem is not in $\mathsf{AC}^0[p^\alpha]$ for odd prime p. To extend the result for the case when $p = 2$ we use the monoid word reduction of matching in bipartite bounded planar cutwidth graphs provided by Hansen et al. [8]. Using this reduction we show that Mod_q can be reduced to perfect matching in bipartite bounded planar cutwidth graphs for some odd prime q. This shows that perfect matching for bipartite bounded planar cutwidth graphs is not in $\mathsf{AC}^0[2^\alpha]$ as well. We also show similar lower bound for series-parallel graphs. An upper bound of NC^1 for perfect matching in series-parallel graphs follows from the result of [6].

1.2 Organization of the Paper

The rest of the paper is organized as follows. In Section 2 we will cover the preliminaries and notations that we will be using throughout the paper. We also discuss the work of Thérien and Barrington [2] and results from Hansen et al. [8]. In Section 3 we show the reduction of Parity to perfect matching in bipartite bounded planar cutwidth graphs. In Section 4 we first discuss the reduction framework of BGGM to the monoid word problem due to Hansen et al. [8]. We then use this framework to show that perfect matching in bipartite bounded planar cutwidth graphs in not in $\mathsf{AC}^0[2^\alpha]$. In Section 5 we discuss an application of our result to perfect matching in series-parallel graphs. We also discuss possible limitations of our approach and future directions.

2 Preliminaries

In this section, we give the required definitions and notations that we use in this paper. We also state the results from previous work that we use in our paper.

2.1 Definitions and Notations

Circuits are a non-uniform model of computation where size and depth of the circuit are two common resources that are usually studied. Additionally, type of gates used in the circuit and fan-in (indegree of a gate) are also often considered. AC^0 is the class of problems having a family of circuits that have the constant depth, polynomial size and unbounded fan-in AND, OR and NOT gates. $\mathsf{AC}^0[m]$ is an extension of AC^0 where the circuits are allowed to have Mod_m gates in addition to AND, OR and NOT gates. ACC^0 is an extension of $\mathsf{AC}^0[m]$ where circuits are allowed to have Mod_m gates for any $m \in \mathbb{N}$. NC^1 is the class of problems having a family of circuits that have logarithmic depth, polynomial size and bounded fan-in AND, OR and NOT gates. It is easy to see that $\mathsf{AC}^0 \subseteq \mathsf{AC}^0[m] \subseteq \mathsf{ACC}^0 \subseteq \mathsf{NC}^1$. In fact, the first containment is proper. The reader can refer to the book by Vollmer for more details about these classes and circuit complexity in general [15]. Let $N_a(x)$ be number of times symbol a appears in string x. We also consider the language $\mathsf{Parity} = \{x \in \{0,1\}^* \mid N_1(x) \not\equiv 0 \mod 2\}$ and its generalization $\mathsf{Mod}_p = \{x \in \{0,1\}^* \mid N_1(x) \not\equiv 0 \mod p\}$ for any $p \geq 2$, for proving our lower bounds.

A *monoid* $\mathcal{M}$ is a set S along with a binary operator $\oplus$ such that (i) for all $s_1, s_2 \in S$, $s_1 \oplus s_2 \in S$ (closure property), (ii) for all $s_1, s_2, s_3 \in S$ we have $s_1 \oplus (s_2 \oplus s_3) = (s_1 \oplus s_2) \oplus s_3$ (associativity property) and (iii) there exists $e \in S$ such that for all $s \in S$ we have $e \oplus s = s = s \oplus e$ (existence of identity). A subset $\mathcal{G}$ of $\mathcal{M}$ is a group in $\mathcal{M}$ if $\mathcal{G}$ is a group with respect to the operation of $\mathcal{M}$. If every group in a monoid is trivial then the monoid is said to be an *aperiodic monoid*. If every group in a monoid is solvable then the monoid is said to be a *solvable monoid*. For a monoid $\mathcal{M}$, the *monoid word problem* is given $x_1, x_2, \ldots, x_n \in \mathcal{M}$ as input, to compute $x_1 \oplus x_2 \oplus \ldots \oplus x_n$.

A *grid graph* is a graph G embedded in an integer lattice such that each edge is either horizontal or vertical. A *grid layered planar graph* is a planar graph G embedded in an integer lattice such that if there is an edge between (a, b) and (c, d) then $|a - c| \leq 1$. Length and width of a grid layered planar graph are the number of columns and rows in the graph respectively.

For a linear arrangement of vertices of a graph G, the maximum number of edges cut by any vertical line is called cutwidth of the linear arrangement of G. *Cutwidth* of G is the minimum cutwidth of a linear arrangement over all possible linear arrangements of G. For a linear arrangement of vertices of a graph G without edge crossings, the maximum number of edges cut by any vertical line is called planar cutwidth of the linear arrangement. *Planar cutwidth* of G is the minimum planar cutwidth of a linear arrangement over all possible planar linear arrangements of G. If a planar linear arrangement of G is not possible, for example in non-planar graphs, we define planar cutwidth of G to be infinite. Note that planar cutwidth of planar graphs is not same as cutwidth of planar graphs.

Graphs with bounded planar cutwidth can be converted into grid layered planar graph preserving matching and bipartiteness. Since constructing such an embedding is not known to be in NC^1 and supposed to be hard for NC^1, we will assume that input is provided as a bipartite grid layered planar graph. This assumption on input is also made by Hansen et al. [8]. Here we consider the circuit complexity of perfect matching in bipartite grid layered planar graphs. Formally BGGM is the set of instances of bipartite grid layered planar graphs along with their embeddings such that they have a perfect matching.

2.2 Algebraic Characterization of Classes in NC^1

We start by describing the definition of bounded width polynomial size programs over monoids as given in [1]. An *instruction* I over monoid $\mathcal{M}$ is a 3-tuple $\langle j, a_0, a_1\rangle$ where $j \in \mathbb{N}$ and $a_0, a_1 \in \mathcal{M}$. For some string $x \in \{0,1\}^*$, we define $I(x) = a_{x_j}$. For some $n \in \mathbb{N}$, a *bounded width polynomial size branching program* (in short BWBP) is a tuple of polynomial number of instruction over some finite monoid $\mathcal{M}$. If $P = (I_1, I_2, \ldots, I_l)$, then for all strings $x \in \{0,1\}^n$, $P(x) = \prod_{i=1}^{l} I_i(x)$ where I_i's are instructions over the monoid $\mathcal{M}$, l is a polynomial in n and product is the operation over monoid. Given an accepting set $\mathcal{A} \subseteq \mathcal{M}$, we say a program P recognizes string x if and only if $P(x) \in \mathcal{A}$. We say a language L is recognized by a family of BWBP, $\langle P_n\rangle$ if and only if P_n recognizes exactly the set of all length n strings in L.

In a seminal work in 1986, Barrington gave the following characterization of NC^1.

Theorem 1. *[1] A language L is in NC^1 if and only if L is recognized by a family of BWBP over some finite monoid.*

In the following year Barrington and Thérien extended their characterization to other subclasses in NC^1.

Theorem 2. *[2] For a language L we have,*

1. *L is in AC^0 if and only if L is recognized by a family of BWBP over an aperiodic finite monoid,*

2. *L is in $\mathsf{AC}^0[p^\alpha]$ for a prime p and constant α if and only if L is recognized by a family of BWBP over a solvable finite monoid in which all groups have order that divide power of p, and,*

3. *L is in ACC^0 if and only if L is recognized by a family of BWBP over a solvable finite monoid,*

Part 2 of Theorem 2 is not directly stated in [2] but can be derived from their proof as also claimed in [8]. We will use these results to show that BGGM is not in $\mathsf{AC}^0[p^\alpha]$ where p is prime and $\alpha \in \mathbb{N}$.

3 BGGM is as hard as Parity

In this section we will give an AC^0 reduction from Parity to BGGM.

Let $x = x_1 x_2 \ldots x_n \in \{0,1\}^*$ be an instance of Parity. Define a function $f(x)$ as

$$f(x) = 0\,\mathrm{bd}(0x_10x_20\ldots0x_n0)0.$$

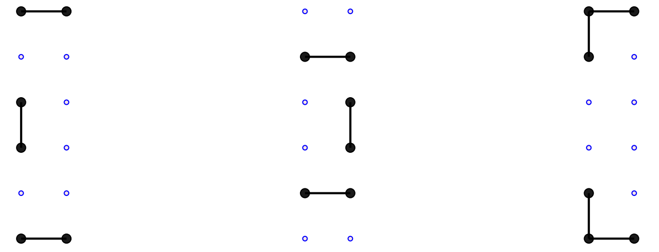
where bd is the bit-double function defined as $\mathrm{bd}(y_1y_2\dots y_n) = y_1y_1y_2y_2\dots y_ny_n$. Clearly, f is an AC^0 computable function. Note that $f(x)$ always has even length and we can visualize $f(x)$ as concatenation of pairs of 2 bits. That is, the first pair contains the first and second bits of $f(x)$, second pair contains the third and fourth bits of $f(x)$ and so on. We will call these pairs as *constituent pairs* of $f(x)$. We note some properties of $f(x)$ that can easily be verified.

Theorem 1. *For every string $x \in \{0,1\}^*$,*

- 11 *cannot be a constituent pair of $f(x)$, and*
- *in $f(x)$, a constituent pair* 01 *is always succeeded by the constituent pair of* 10 *and a constituent pair* 10 *is always preceded by the constituent pair* 01*.*

Using $f(x)$ we construct a bipartite grid layered planar graph G_x, such that, G_x has a perfect matching if and only if x has even parity. Also, we will show that G_x can be constructed from $f(x)$ in AC^0. This will imply that Parity reduces to BGGM.

First we define graph blocks G_{00}, G_{01} and G_{10} corresponding to the three constituent pairs 00, 01 and 10 respectively as shown in Figure 1. Note that 11 cannot be a constituent pair hence we do not define a graph corresponding to it. These graph blocks will be the constituent elements of the graph G_x.



(a) G_{00} (b) G_{01} (c) G_{10}

Figure 1: Different types of Graph Blocks

We also define an operator $\odot$ over these graphs which allows us to define larger graphs using these graph blocks. $\odot$ operator is defined in Figure 2.

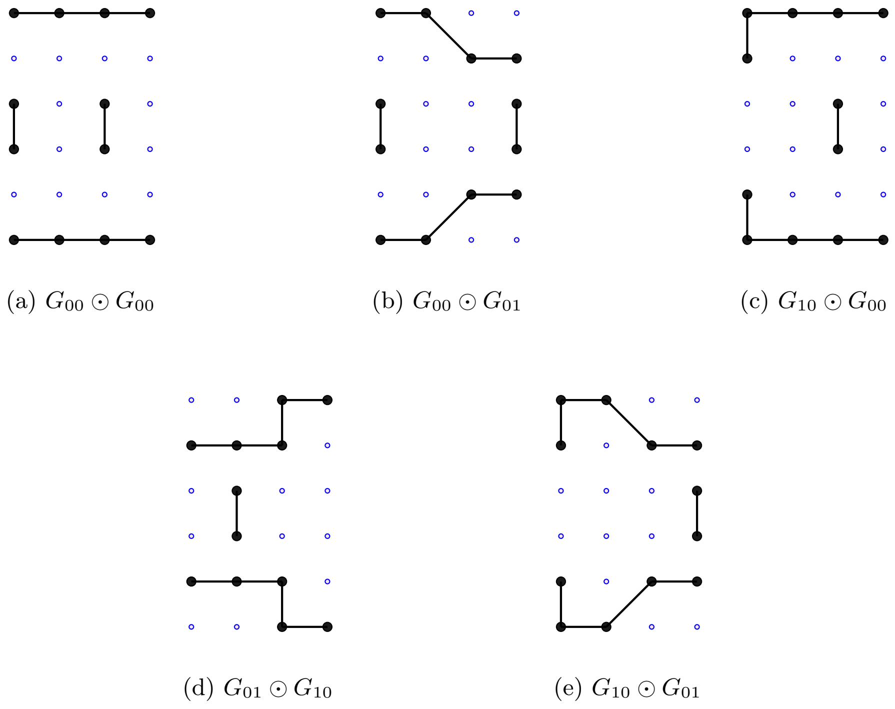
We now complete the construction of G_x. Let $y = f(x) = y_1y_2\dots y_m$. where m is even. Then

$$G_x = G_{y_1y_2} \odot G_{y_3y_4} \odot \dots \odot G_{y_{m-1}y_m}.$$

For example if $x = 1101$, then $f(x) = 0bd(010100010)0 = 00011001100000011000$ and G_x will be as shown in Figure 3.

Theorem 2. *For every string $x \in \{0,1\}^*$, G_x is a bipartite grid layered planar graph. Also, each connected component of G_x is either a single edge or a path that extends from the first block to the last block of G_x.*

Proof. By definition, each of the three graph blocks is grid layered planar graphs. Moreover, the operator $\odot$ connects adjacent blocks by preserving planarity and the overall grid structure. Hence G_x is a grid layered planar graph.


(a) $G_{00} \odot G_{00}$

(b) $G_{00} \odot G_{01}$

(c) $G_{10} \odot G_{00}$

(d) $G_{01} \odot G_{10}$

(e) $G_{10} \odot G_{01}$

Figure 2: Joining different Blocks with $\odot$ operation

To show that each connected component of G_x is a path we will use induction on the number constituent pairs of $y = f(x)$. For the base case note that if y has only one constituent pair then it must be 00 and G_{00} contains only paths of even length (number of vertices). Now consider a graph G_p corresponding to the first $m-1$ constituent pairs of y. Assume that every connected component in G_p is a path. Let $G_{p'}$ be the graph corresponding to the first m constituent pairs of y. If the last block of G_p is G_{00} then by Lemma 1, the next block can either be G_{00} or G_{01}. By Figure 2a and 2b we have that $G_{p'}$ will only be extending the paths of G_p in addition to two isolated edges. So every connected component in $G_{p'}$ will be a path as well. Similarly if the last block of G_p is G_{01} then again by Lemma 1, the next block will be G_{10} and by Figure 2d we have that every connected component in $G_{p'}$ will be a path as well. Finally if the last block of

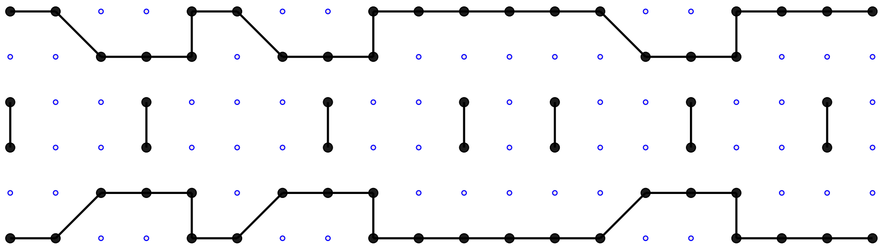


Figure 3: Graph G_x corresponding to the string $x = 1101$

G_p is G_{10} then the next block can either be G_{00} or G_{01} and by Figure 2c and 2e we have that every connected component in $G_{p'}$ will be a path as well. Also, note that each path in $G_{p'}$ extends from the first block to the last one or is of length one.

This also shows that G_x is bipartite since it does not have any cycles. □

Theorem 3. *For every string $x \in \{0,1\}^*$, G_x has a perfect matching if and only if x has even parity.*

Proof. We claim that G_x has a perfect matching if and only if it has an even number of G_{10} blocks. Since the number of G_{10} blocks in G_x is same as the number of ones in x, this will complete the proof. To prove our claim we again use induction on the number constituent pairs of y.

For the base case note that G_{00} has a perfect matching using all its three edges. Assume we have a graph G_p corresponding to the first $m-1$ constituent pairs of y such that G_p has a perfect matching if and only if G_p has an even number of G_{10} blocks. Now suppose we are extending the graph G_p by one graph block to get the graph $G_{p'}$. We divide this into two cases.

Case 1: $G_{p'} = G_p \odot G_{00}$ or $G_{p'} = G_p \odot G_{01}$. In this case G_p and $G_{p'}$ have the same number of G_{10} blocks. By construction two paths in $G_{p'}$ get extended by two vertices while others remain the same. Also, an additional new edge is introduced whose endpoints are matched with each other (see Figure 2). Thus if G_p has a perfect matching then $G_{p'}$ will also have a perfect matching where the two new vertices by which the paths get extended, are matched with each other. If G_p does not have a perfect matching then at least one of its paths has an odd number of vertices. Extending this path by two more vertices preserves its parity and hence $G_{p'}$ will also not have a perfect matching.

Case 2: $G_{p'} = G_p \odot G_{10}$. In this case $G_{p'}$ has an extra G_{10} block from G_p. Two paths in $G_{p'}$ get extended by three vertices while others remain the same (see Figure 2d). Also, note that each path is symmetric about a horizontal axis passing through the centre. Therefore both paths of length more than one has the same length. If G_p has a perfect matching then the new graph $G_{p'}$ does not have perfect matching as the number of vertices in the two paths become odd and hence cannot be matched. On the other hand, if G_p does not have a perfect matching then both the long paths have an odd number of vertices. Hence in $G_{p'}$ these paths will have an even number of vertices and hence a perfect matching exists in $G_{p'}$.

□

Theorem 4. *For every string $x \in \{0,1\}^*$, G_x and its planar embedding is computable in AC^0.*

Proof. It is easy to see that G_x can be computed using AC^0 circuits when $f(x)$ is given as input as each vertex depends on at most two bits of $f(x)$ and each edge depends on at most four bits of $f(x)$. Also the embedding is AC^0-computable as we can compute whether (a,b) is a vertex using just two bits of $f(x)$ and whether (a,b) and (c,d) have edge between them using just four bits of $f(x)$. Next we show that G_x and its planar embedding is AC^0-computable even when

x is given as input. For this note that each bit of $f(x)$ is either 0 or a copy of some bit in x. In circuit taking y as input, we can hardcode some bits to 0 and pass input from bits of x wherever copy bit of some bit in x are used. Thus, we get a circuit which computes G_x and its planar embedding using x as input. □

Theorem 3. Parity *reduces to* BGGM *in* AC^0.

Theorem 3 follows from Lemmas 2, 3 and 4. Razborov and Smolensky had independently shown the following result.

Theorem 4. *[12, 13] Let p and q be two distinct prime numbers and $\alpha \in \mathbb{N}$.* $\mathsf{Mod}_q \notin \mathsf{AC}^0[p^\alpha]$.

Now by combining the result of Razborov and Smolensky and applying Theorem 3 we have the following result.

Theorem 5. BGGM *is not in* $\mathsf{AC}^0[p^\alpha]$ *for every odd prime p and $\alpha \in \mathbb{N}$.*

4 Lower Bounds for BGGM

Now we will show that BGGM is not in $\mathsf{AC}^0[2^\alpha]$ as well. For this, we will first describe reduction given in [8] that shows BGGM is in ACC^0.

4.1 Reduction of BGGM to Monoid Word Problem

For two relations $R, S \subseteq \mathcal{A} \times \mathcal{B}$. We define $RS = \{(x, y) \mid \exists z$ such that $(x, z) \in R$ and $(z, y) \in S\}$. For any grid layered planar graph G, we will first define the monoid element corresponding to G. We will denote this monoid element by $G^{\mathcal{M}}$. For any G, $G^{\mathcal{M}} = (X, Y, R)$ where X is set of vertices in leftmost layer of G, Y is set of vertices in rightmost layer of G and $R \subseteq 2^X \times 2^Y$ is a relation. Let $X' \subseteq X$ and $Y' \subseteq Y$ then $(X', Y') \in R$ if and only if $G \setminus (\overline{X'} \cup Y')$ has a perfect matching. In other words, there is a matching in G which matches every vertex except those in $\overline{X'}$ and Y'. We now define the monoid as the set

$$\mathcal{M} = \{G^{\mathcal{M}} \mid G \text{ is a bipartite grid layered planar graph}\} \cup \{0, 1\}$$

where 1 is the identity element of the monoid and the operation of the monoid (denoted as $*$) is defined as

$$(W, X, R) * (Y, Z, S) = \begin{cases} (W, Z, RS) & \text{if } X = Y \\ 0 & \text{if } X \neq Y \end{cases}$$

and for all $M \in \mathcal{M}$, $0 * M = M * 0 = 0$. For the remaining part of Section 4 we will refer to this monoid as $\mathcal{M}$.

Next, we define a concatenation operation on grid layered planar graphs. Let G_1 and G_2 be two grid layered planar graphs having same width w and lengths l_1 and l_2 respectively. We define $G_1 \cdot G_2$ to be the grid layered planar graph having width w and length $l_1 + l_2$, obtained by identifying the vertices in the rightmost column of G_1 and the leftmost column of G_2. Here we assume that there are no vertical edges present in the leftmost or rightmost column of a grid layered planar graph. This can be assumed without loss of generality because given any grid layered planar graph we can convert it to a grid layered planar graph having the above property by adding additional columns to the left and right, and adding edges appropriately such that it preserves matching. Then we have the property that $(G_1 \cdot G_2)^{\mathcal{M}} = {G_1}^{\mathcal{M}} * {G_2}^{\mathcal{M}}$.

4.2 BGGM is not in $\mathsf{AC}^0[2^\alpha]$

In this section we show that BGGM is also not in $\mathsf{AC}^0[2^\alpha]$. First we will show an algebraic property of the monoid $\mathcal{M}$ defined in Section 4.1.

Theorem 5. *There exists a cyclic group $\mathcal{G}$ in $\mathcal{M}$ of order p where p is an odd prime.*

Proof. Since by Theorem 3 BGGM is not in AC^0, hence $\mathcal{M}$ is not an aperiodic monoid by the characterisation provided in [2]. Thus a group $\mathcal{G}_n \subseteq \mathcal{M}$ and $\mid \mathcal{G}_n \mid = n > 1$ exists. Hansen et al. showed that every group contained in $\mathcal{M}$ will have odd order [8]. Thus, some odd prime $p|ord(\mathcal{G}_n)$. Thus there exist a cyclic subgroup of $\mathcal{M}_n$, $\mathcal{G}$, of order p. □

Theorem 6. *Let $\mathcal{G}$ be a cyclic group of order p in $\mathcal{M}$ and e is the identity of $\mathcal{G}$. Then for some generator of $\mathcal{G}$, say x, there exists grid layered planar graphs A and B such that $A^{\mathcal{M}} = x$, $B^{\mathcal{M}} = e$ and A and B have same length.*

Proof. We have $x, e \in \mathcal{M}$ such that $x \neq e$. Moreover it is easy to see that the elements 0 and 1 of $\mathcal{M}$ are not contained in $\mathcal{G}$. Hence there exists grid layered planar graphs A' and B' such that $A^{\mathcal{M}} = x$ and $B^{\mathcal{M}} = e$. If A and B have same length we are done. Otherwise using A and B we will give the construction of grid layered planar graphs A' and B' such that lengths of A' and B' are same, $A'^{\mathcal{M}} = y$ and $B'^{\mathcal{M}} = e$ where y is a generator of $\mathcal{G}$. Consider the following cases:

Case 1: Difference between the lengths of A and B is even. Without loss of generality assume A has smaller length. We construct A' by adding an even number of columns to the right hand side of A such that A' and B have the same number of columns. Now we add horizontal paths of even length from each vertex in the rightmost column of A to its corresponding vertex in the rightmost column of A'. Since we have added paths of even length, therefore there there is a one to one correspondence between matchings in A and A'. Hence $A'^{\mathcal{M}} = A^{\mathcal{M}}$.

Case 2: A has odd length and B has even length. As $B^{\mathcal{M}} = e$ we have $(B \cdot B)^{\mathcal{M}} = e^2 = e$. Also $B \cdot B$ will have odd length. Hence it reduces to Case 1.

Case 3: A has even length and B has odd length. Note that if $A^{\mathcal{M}} = x$ has order p then $(A \cdot A)^{\mathcal{M}} = x^2$ also have order p. Now $A \cdot A$ has odd length. Hence it reduces to Case 1 again.

□

Consider the grid layered planar graphs A and B as obtained by Lemma 6. Using them we define a function h from the set of all binary strings to the set of all grid layered planar graphs. h is defined recursively as follows:

$$\begin{aligned} h(\epsilon) &= B \\ h(y0) &= h(y) \cdot B \\ h(y1) &= h(y) \cdot A \end{aligned}$$

Note that $h(z)$ is essentially the grid layered planar graph obtained by concatenating copies of A and B for every 1 and 0 in the string z respectively, together with an extra B for ϵ at the leftmost end.

Theorem 7. *h is AC^0-computable.*

Proof. By Lemma 6, for every positive integer n there exists grid layered planar graphs A and B such that $A^{\mathcal{M}} = x$, $B^{\mathcal{M}} = e$ and A and B have the same length (say m) and same width (say w). We assume that A and B are hardcoded into the AC^0 circuit say C_n. Now given a string $z \in \{0,1\}^n$, for every bit 0 or 1 of z, the circuit C_n outputs the corresponding graph B or A respectively in the order of the input bits. Additionally C_n also outputs a copy of the graph B at the beginning. Hence the output graph will have width w and length $m + n(m-1)$. Here we will crucially use the fact that A and B have same the length, since otherwise the length of the output graph would have been variable. □

Theorem 8. *Let $\mathcal{G}$ be the group as obtained in Lemma 5 and let e be the identity element in $\mathcal{G}$. For all strings $z \in \{0,1\}^*$, $z \in \mathsf{Mod}_p$ if and only if $h(z)^{\mathcal{M}} \neq e$.*

Proof. Let $z = z_1 z_2 \ldots z_n$ such that $z_i \in \{0,1\}$. Then,

$$\begin{aligned} h(z)^{\mathcal{M}} &= h(z_1 z_2 \ldots z_n)^{\mathcal{M}} \\ &= (B \cdot X_1 \cdot X_2 \cdot \ldots \cdot X_n)^{\mathcal{M}}, \text{ such that } X_i = A \text{ if } z_i = 1 \text{ and } X_i = B \text{ if } z_i = 0 \\ &= B^{\mathcal{M}} * X_1^{\mathcal{M}} * X_2^{\mathcal{M}} * \ldots * X_n^{\mathcal{M}} \\ &= x^t, \text{ where } t \text{ is the number of 1's in } z \end{aligned}$$

Now by Lemma 5 we have $x^t = e$ if and only if $t \equiv 0 \mod p$. Hence $z \in \mathsf{Mod}_p$ if and only if $h(z)^{\mathcal{M}} \neq e$. □

Theorem 9. *Consider the language $L = \{h(z) \mid h(z)^{\mathcal{M}} = e$ where $z \in \{0,1\}^*\}$. Then L reduces to BGGM in AC^0.*

Proof. We know that $h(z)$ is a grid layered planar graph having fixed width, say w. Let $e = (X, Y, R)$ and $h(z)^{\mathcal{M}} = (X_0, Y_0, R_0)$. Assume $e = (X, Y, R)$ are hardcoded in the circuit. We can easily check if $X_0 = X$ and $Y_0 = Y$ using an AC^0 circuit.

For checking whether $R_0 = R$, we will create 2^w instances of BGGM and infer R_0 from their output. For each $X' \subseteq X$ and $Y' \subseteq Y$, we create a graph $G_{X'Y'}$. $G_{X'Y'}$ is the graph $h(z)$ with some additional vertices and edges. For each $v \in \overline{X'}$ we add a vertex l_v to the left of v and the edge $\{v, l_v\}$. Similarly, For each $v \in Y'$ we add a vertex r_v to the right of v and the edge $\{v, r_v\}$. Note that $(X', Y') \in R_0$ if and only if $G_{X'Y'}$ have a perfect matching. Thus, we can compute R_0. Now R_0 if and only if for all $X' \subseteq X$ and $Y' \subseteq Y$ we have $G_{X'Y'}$ contains a perfect matching. So our output graph is essentially a union of all such graphs $G_{X'Y'}$. Again this can be constructed easily in AC^0. □

Theorem 6. $\overline{\mathsf{Mod}_p}$ *reduces to* BGGM *in* AC^0.

Proof. Given $z \in \{0,1\}^n$ we first compute $h(z)$ using Lemma 7 and then output the grid layered planar graph (say G_z) as obtained by the reduction of Lemma 9. By Lemma 8 and 9 it follows that $z \notin \mathsf{Mod}_p$ if and only G_z has a perfect matching. □

Theorem 7. BGGM *is not in* $\mathsf{AC}^0[2^\alpha]$ *for* $\alpha \in \mathbb{N}$.

Proof. If BGGM is in $\mathsf{AC}^0[2^\alpha]$ then by combining this circuit with the reduction in Theorem 6 and appending a NOT gate at the top we would get an $\mathsf{AC}^0[2^\alpha]$ circuit for Mod_p. This would contradict Razborov and Smolensky's result (Theorem 4) since p is an odd prime. □

Finally combining Theorems 5 and 7 we get the following theorem.

Theorem 8. BGGM *is not in* $\mathsf{AC}^0[p^\alpha]$ *for every prime* p *and* $\alpha \in \mathbb{N}$.

5 Application of our results

5.1 Circuit Lower Bounds for Series-Parallel Graphs

For series-parallel graphs, it is known that bipartite matching is in NC^1 given transitive closure of tree decomposition as input as well [5]. We show a better lower bound for this problem using our reduction of Parity to BGGM.

In reduction from Parity to BGGM the final graph we get is also a series-parallel graph (each connected component is a path or a single edge). Now the challenge is to construct the transitive closure of the tree decomposition(more precisely term representation of tree decomposition) for it using AC^0 circuits. For this, we will use the following theorem mentioned in Hansen et al. [8].

Theorem 9. *[8] Given as input a linear arrangement of bounded cutwidth k for some graph G, a tree decomposition of width k for graph G in term representation can be constructed by an* AC^0 *circuit.*

Theorem 10. *For every $x \in \{0,1\}^*$, a linear arrangement of bounded cutwidth for G_x can be created by an* AC^0 *circuit.*

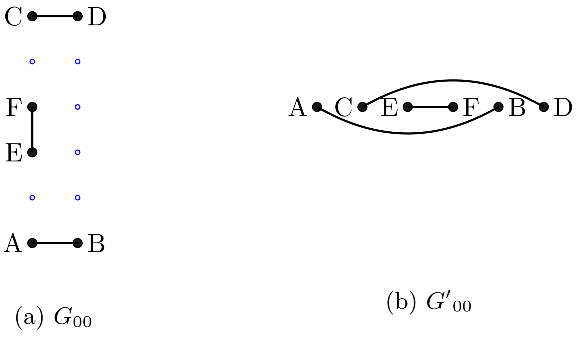
Proof. We linearize G_{00}, G_{01} and G_{10}. For G_{00} it is shown in figure 4. Same can be done for G_{01} and G_{10}. This order can be hardcoded in AC^0 circuit. We keep edges same. Clearly to check edge between v_i and v_j, we will not need more than 4 bits. It can be shown that such a linear arrangement can be computed in AC^0 using arguments similar to those given in Section 3. □

Theorem 10. Parity *reduces to series-parallel graph matching with given tree decomposition in term representation.*

Proof. Follows from Theorem 9 and Lemma 10. □

Theorem 11. *Series-parallel graph matching with given tree decomposition in term representation is not in* $\mathsf{AC}^0[p^\alpha]$ *where p is odd prime and $\alpha \in \mathbb{N}$.*

Proof. Follows from Theorems 10 and 4. □



(a) G_{00}

(b) G'_{00}

Figure 4: Linearizing G_{00}

5.2 Future Work

For $m \in \mathbb{N}$ and not a power of prime we do not know whether $\mathsf{AC}^0[m] = \mathsf{NP}$. To extend our results for all $m \in \mathbb{N}$ or to show that BGGM lies in $\mathsf{AC}^0[m]$ for some $m \in \mathbb{N}$, algebraic characterization of these classes are needed. Algebraic theory of subclasses of NC^1 developed by [2] does not provide any such characterization. This is the biggest hurdle in extending our approach to $\mathsf{AC}^0[m]$.

Our result improves the lower bound for perfect matching in series-parallel graphs but the bound are not tight. For example, we do not know if perfect matching in series-parallel graphs is in $\mathsf{AC}^0[2]$ or ACC^0.

Acknowledgment

The authors would like to thank Sarthak Garg for several sessions of helpful discussions. The second author would like to acknowledge Navid Talebanfard for introducing him to the problem. He would also like to thank Jayalal Sarma for inviting him to the PhD defence of Balagopal Komarath where he gained more insight into the area in general.

References

[1] D A Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC1. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, STOC '86, pages 1–5, New York, NY, USA, 1986. ACM. URL: `http://doi.acm.org/10.1145/12130.12131`, `doi:10.1145/12130.12131`.

[2] David A. Mix Barrington and Denis Thérien. Finite monoids and the fine structure of NC1. *J. ACM*, 35(4):941–952, October 1988. URL: `http://doi.acm.org/10.1145/48014.63138`, `doi:10.1145/48014.63138`.

[3] Bireswar Das, Samir Datta, and Prajakta Nimbhorkar. Log-space algorithms for paths and matchings in k-Trees. *Theor. Comp. Sys.*, 53(4):669–689, November 2013. URL: `http://dx.doi.org/10.1007/s00224-013-9469-9`, `doi:10.1007/s00224-013-9469-9`.

[4] J. Edmonds. Paths, trees and flowers. *Canad. J. Math.*, 17:449–467, 1965.

[5] Michael Elberfeld, Andreas Jakoby, and Till Tantau. Logspace versions of the theorems of Bodlaender and Courcelle. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 143–152, Washington, DC, USA, 2010. IEEE Computer Society. URL: http://dx.doi.org/10.1109/FOCS.2010.21, doi:10.1109/FOCS.2010.21.

[6] Michael Elberfeld, Andreas Jakoby, and Till Tantau. Algorithmic meta theorems for circuit classes of constant and logarithmic depth. In Christoph Dürr and Thomas Wilke, editors, *29th International Symposium on Theoretical Aspects of Computer Science, STACS 2012, February 29th - March 3rd, 2012, Paris, France*, volume 14 of *LIPIcs*, pages 66–77. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012. URL: https://doi.org/10.4230/LIPIcs.STACS.2012.66, doi:10.4230/LIPIcs.STACS.2012.66.

[7] Stephen Fenner, Rohit Gurjar, and Thomas Thierauf. Bipartite perfect matching is in quasi-NC. In *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 754–763, New York, NY, USA, 2016. ACM. URL: http://doi.acm.org/10.1145/2897518.2897564, doi:10.1145/2897518.2897564.

[8] Kristoffer Arnsfelt Hansen, Balagopal Komarath, Jayalal Sarma, Sven Skyum, and Navid Talebanfard. *Circuit Complexity of Properties of Graphs with Constant Planar Cutwidth*, pages 336–347. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014. URL: https://doi.org/10.1007/978-3-662-44465-8_29, doi:10.1007/978-3-662-44465-8_29.

[9] R M Karp, E Upfal, and A Wigderson. Constructing a perfect matching is in Random NC. *Combinatorica*, 6(1):35–48, January 1986. URL: http://dx.doi.org/10.1007/BF02579407, doi:10.1007/BF02579407.

[10] László Lovász. On determinants, matchings, and random algorithms. In *FCT*, pages 565–574, 1979.

[11] K. Mulmuley, U. V. Vazirani, and V. V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7(1):105–113, January 1987. URL: http://dx.doi.org/10.1007/BF02579206, doi:10.1007/BF02579206.

[12] Alexander A. Razborov. On the method of approximations. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 167–176, 1989. URL: http://doi.acm.org/10.1145/73007.73023, doi:10.1145/73007.73023.

[13] R. Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, pages 77–82, New York, NY, USA, 1987. ACM. URL: http://doi.acm.org/10.1145/28395.28404, doi:10.1145/28395.28404.

[14] Ola Svensson and Jakub Tarnawski. The matching problem in general graphs is in quasi-NC. *CoRR*, abs/1704.01929, 2017. URL: http://arxiv.org/abs/1704.01929.

[15] Heribert Vollmer. *Introduction to Circuit Complexity: A Uniform Approach.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1999.