Planar Maximum Matching: Towards a Parallel Algorithm

Samir Datta[1]
Chennai Mathematical Institute & UMI ReLaX, Chennai, India
sdatta@cmi.ac.in

Raghav Kulkarni
Chennai Mathematical Institute, Chennai, India
kulraghav@gmail.com

Ashish Kumar
Chennai Mathematical Institute, Chennai, India
ashishky36@gmail.com

Anish Mukherjee[2]
Chennai Mathematical Institute, Chennai, India
anish@cmi.ac.in

Abstract

Perfect matchings in planar graphs have been extensively studied and understood in the context of parallel complexity [21, 36, 25, 6, 2]. However, corresponding results for maximum matchings have been elusive. We partly bridge this gap by proving:

1. An SPL upper bound for planar bipartite maximum matching search.
2. Planar maximum matching search reduces to planar maximum matching decision.
3. Planar maximum matching count reduces to planar bipartite maximum matching count and planar maximum matching decision.

The first bound improves on the known [18] bound of $\mathsf{L}^{\mathsf{C}_{=}\mathsf{L}}$ and is adaptable to any special bipartite graph class with non-zero circulation such as bounded genus graphs, $K_{3,3}$-free graphs and K_5-free graphs. Our bounds and reductions non-trivially combine techniques like the Gallai-Edmonds decomposition [23], deterministic isolation [6, 7, 3], and the recent breakthroughs in the parallel search for planar perfect matchings [2, 32].

2012 ACM Subject Classification Theory of computation → Parallel algorithms

Keywords and phrases maximum matching, planar graphs, parallel complexity, reductions

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2018.21

1 Introduction

Matchings are one of the most fundamental and well-studied objects in graph theory and in theoretical computer science (see e.g. [23, 20]) and have played a central role in Algorithms and Complexity Theory. Edmond's blossom algorithm [8] for Maximum-Matching is one of the first examples of a non-trivial polynomial time algorithm. It has had a considerable share in initiating the study of efficient computation, including the class P itself; Valiant's #P-hardness [35] for counting perfect matchings in bipartite graphs provides surprising

[1] The author was partially funded by a grant from Infosys foundation and SERB grant MTR/2017/000480.
[2] The author was partially supported by a grant from Infosys foundation and TCS PhD fellowship.

insights into counting complexity classes. The rich combinatorial structure of matching problems combined with their potential to serve as central problems in the field invites their study from several perspectives.

We consider the following variants of the Maximum-Matching problem. Given w, the Decision (or Cardinality) version asks to decide if there is a maximum matching of cardinality at least w. The Search and the Counting versions ask respectively for a (witness to a) maximum matching and the number of maximum matchings.

1.1 Parallel Complexity of Matching

The study of whether matching is parallelizable has yielded powerful tools, such as the isolating lemma [29], that have found numerous other applications. The RNC bound remains the best known parallel complexity for Maximum-Matching till date. One of the biggest open problems in this area is to derandomize such construction. Recently, a partial derandomization has put the Perfect-Matching problem in quasi-NC, first for bipartite graphs [10], followed by [34] for general graphs. The best known (non-uniform) upper bound for Perfect-Matching is non-uniform SPL [1].

Matching in Planar and Other Sparse Graphs

A well known example where planarity is a boon is that of counting perfect matchings. The problem in planar graphs is in P [21] and can in fact be placed in NC[36]; thus Perfect-Matching (Decision) in planar graphs is in NC.

In the case of parallel algorithms for planar graphs, the search version seemed harder than the problem of counting. Though the bipartite planar case is known to be in NC[28, 25, 22, 6], the construction version of Perfect-Matching in planar graphs in NC was an outstanding open question and has been solved very recently by Anari and Vazirani [2] and Sankowski [32].

The space complexity of matching problems in planar graphs was first studied in [6] where it is shown that min-weight Perfect-Matching in bipartite planar graphs is in SPL via non-zero circulations. The isolation lemma has also been derandomized for $K_{3,3}$-free and K_5-free bipartite graphs, giving the SPL upperbound [3].

However, known results on Maximum-Matching are limited. The only relevant result known to us is computing a maximum matching for bipartite planar graphs in $\mathsf{L}^{\mathsf{C}_=\mathsf{L}} \subseteq \mathsf{NC}$ by Hoang [18]. A different NC algorithm is given for the same problem in [32]. The related approximation problem has been investigated more. An NC approximation scheme [19] and a Logspace approximation scheme [5] for Maximum-Matching are known for general graphs and classes of sparse graphs (including bounded degree graphs and planar graphs) respectively.

1.2 Maximum Matching and Our Contribution

Since Perfect-Matching is a specialisation of Maximum-Matching, upper bounds applicable for the latter directly translate to the former. Edmond's *blossom-shrinking* algorithm and the Micali-Vazirani [27] algorithm fall in this category. Occasionally, it is possible to lift the bounds in the other direction also such as the following:

▶ **Observation 1.** *Perfect-Matching and Maximum-Matching are equivalent in general graphs under logspace Turing reduction.*

Though if we start with a planar graph such reductions does not necessarily keep the graph planar and the goal of this paper is to explore such possibilities for special graph classes.

Recent years have seen considerable progress in upper bounds for Perfect-Matching in planar and other restricted graph classes culminating in [2, 32] which yield efficient parallel algorithms for planar Perfect-Matching Search. In this paper we try to close the gap between perfect and maximum matchings for planar and related graph classes in the context of parallel complexity. Unless otherwise stated, our results hold in planar, bounded genus, $K_{3,3}$-free and K_5-free graphs. Our main result is the following.

▶ **Theorem 2.** *Maximum-Matching Search in graphs with non-zero circulation is in SPL.*

The class SPL, prominently studied in [1], consists of languages whose characteristic function is computed by a determinant. The bound improves on the best known upper bound of $\mathsf{L}^{\mathsf{C_=L}}$ by Hoang [18] and matches the known upper bound for bipartite Perfect-Matching for the same classes of graphs [6, 3]. Hoang uses a rank argument whose complexity doesn't seem to be in SPL - the seemingly best bound being $\mathsf{L}^{\mathsf{C_=L}}$. Instead, we use the standard isolation technique but in a multi-graph (i.e. with self-loops) but make sure that the *loop-paths* are never optimal and we can focus on the min-weight cycle covers which deterministic isolation helps us find. Since, a deterministic construction of non-zero circulation is known for $K_{3,3}$-free and K_5-free bipartite graphs [3] and bounded genus bipartite graphs [7], the result holds for these classes also.

Next, we reduce the problem of finding a maximum matching to determining the size of a maximum matching in the presence of algorithms to (a) find a perfect matching and to (b) solve the bipartite version of the maximum matching, all in the same class of graphs. We use the classic Gallai-Edmonds decomposition theorem for this reduction. Since NC algorithms are now known for Perfect-Matching in bounded genus [2], $K_{3,3}$-free and K_5-free graphs [9] then in these classes of graphs using Theorem 2 we get the following:

▶ **Theorem 3.** *Maximum-Matching Search NC-reduces to Maximum-Matching Decision in planar graphs, in bounded genus graphs, in $K_{3,3}$-free graphs and in K_5-free graphs.*

This shows that, unlike for perfect matching where decision was known to be in NC and the main bottleneck was the search version, for maximum matching the decision problem is the hard cornerstone. Though we are not able to get an NC upper bound for Maximum-Matching, we show that Maximum-Matching Search for the above mentioned classes of graphs is in *Pseudo-deterministic* NC. Pseudo-deterministic algorithms are probabilistic algorithms for search problems that produce a unique output for each given input except with small probability. That is, they return the same output for all but few of the possible random choices. We call an algorithm pseudo-deterministic NC if it is in RNC, and is pseudo-deterministic. Bipartite Perfect-Matching is known to be in this class [14].

The class of search problems that can be solved in pseudo-deterministic polynomial time was first studied by Goldwasser and Gat [12]. Since then the field of pseudo-determinism has received significant interest, see e.g. [12, 13, 16] with some very recent progress e.g. [31, 14, 17, 15]. As the size of the maximum matching can be found in RNC [29], from Theorem 3 we get that,

▶ **Theorem 4.** *Maximum-Matching Search is in pseudo-deterministic NC for planar graphs, bounded genus graphs, $K_{3,3}$-free graphs and K_5-free graphs.*

We also consider the counting version of the Maximum-Matching problem. Though we don't have an NC algorithm even in planar graphs (in fact, to the best of our knowledge it is not even known to be in P), we show that counting maximum matchings in planar graphs NC-reduces to the question in bipartite planar graphs.

▶ **Theorem 5.** *Maximum-Matching Count* NC*-reduces to Bipartite-Maximum-Matching Count and Maximum-Matching Decision in planar, bounded genus, $K_{3,3}$-free and K_5-free graphs.*

The main questions left unanswered in this study are:

▶ **Open Question 1.** *Is Maximum-Matching Decision in planar graphs in* NC*?*

▶ **Open Question 2.** *Is Maximum-Matching Count in bipartite planar graphs in* NC*?*

Organization. After some preliminaries in Section 2, we describe in Section 3 the SPL algorithm for finding maximum matchings in graphs that have non-zero circulations. For bounded genus, $K_{3,3}$-free and K_5-free graphs, we the give an NC-reduction from the problem of finding a maximum matching to determining the size of a maximum matching, in Section 4. In Section 5, for the same graph classes we show that counting maximum matching NC-reduces to counting maximum matchings in bipartite graphs and determining the size of a maximum matching. We conclude in Section 6 with some open ends.

2 Preliminaries

Let $G = (V, E)$ be an undirected embedded planar graph with $|V| = n$. We sometime think of the edges as bi-directed i.e. they are directed in both the directions. For $e \in E$, let $w(e)$ denote the weight of the edge e. A planar graph is a graph that can be embedded in the plane so that no edges cross each other. A graph G is said to have *genus* g if G has a minimal embedding (an embedding where every face of G is homeomorphic to a disc) on a genus g surface. An H-minor free graph G does not contain the graph H as a minor. See standard texts on Graph theory (e.g. [37]) for further information. Consult [30] for definitions and properties of various other sparse graph classes.

A matching in G is a set $M \subseteq E$, such that no two edges in M have a vertex in common. A matching M is called *perfect* if M covers all vertices of G, M of maximum size is called *maximum* matching. An alternating path is one whose edges alternate between M and $E \setminus M$. We denote the size (the number of edges in the matching) of M by $|M|$ and the weight (sum of the weight of the edges in the matching) by $w(M)$. Size of the maximum matching in G is denoted by $\nu(G)$. We call two edges (also self-loops and multiple edges) of G' disjoint, if the set of vertices which are incident on the edges are disjoint. A matching M of G is said to be *near-perfect* if exactly one vertex of G is not matched in M. For a complete treatment on matching see [23].

Complexity Classes. The complexity classes L and NL are the classes of languages accepted by deterministic and non-deterministic logspace Turing machines, respectively. For a non-deterministic Turing machine M, let $acc_M(x)$ and $rej_M(x)$ denote the number of accepting and rejecting computations respectively, on an input x. Denote $gap_M(x) = acc_M(x) - rej_M(x)$. GapL is the class of functions $f(x)$ such that for some NL machine M, $f(x) = gap_M(x)$. A language L is in SPL if so that for all inputs $x, gap_M(x) \in \{0, 1\}$ and $x \in L$ if and only if $gap_M(x) = 1$. For a complexity class $\mathcal{C}$, we say that a language L *$\mathcal{C}$-reduces* to a language L' if there is a many-one reduction from L to L' computable in the class $\mathcal{C}$. NC (RNC) is the class of problems which can be solved using deterministic (randomized) polynomial size circuits of polylogarithmic depth. Define pseudo-deterministic algorithms as follows:

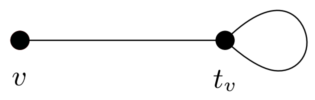

Figure 1 Gadget added at each vertex $v \in G$ to construct G'.

▶ **Definition 6** ([14]). An algorithm A for a relation R is *pseudo-deterministic* if there exists some function s such that A, when executed on input x, outputs $s(x)$ with high probability, and s satisfies $(x, s(x)) \in R$.

A *pseudo-deterministic NC* algorithm is an RNC algorithm which is also pseudo-deterministic.

Non-zero Circulation Weights. For any simple cycle C of G, we define the *circulation* of C, denoted by $\mathrm{circ}(C)$, as the alternating sum of the weights of the edges of the cycle. Formally, if the cycle is given by $(e_0, e_1, \ldots e_\ell)$ then, $\mathrm{circ}(C) = \sum_{i=0}^{\ell}(-1)^i w(e_i)$. In this paper the classes of graphs, where deterministic weighting schemes are known such that each cycle in the given graph gets non-zero circulation weight, are together referred to as *graphs with non-zero circulation*.

It is shown in [6] that non-zero circulation weights imply isolating weights for matchings. Also, a simple L-computable weighting function is constructed for grid graphs such that the circulation of every simple cycle is non-zero. In [7] it is shown that using [4] this weighting function can be extended to all bipartite graphs embeddable on a fixed surface. This was further extended to $K_{3,3}$-free and K_5-free bipartite graphs in [3].

3 Maximum-Matching Search in graphs with non-zero circulations

In this section we show that given an undirected unweighted graph $G = (V, E)$ admitting non-zero circulations, finding a maximum matching is in SPL. The basic idea is to construct an auxiliary graph G' having the property that finding a maximum matching in G reduces to finding a min-weight generalized perfect matching (defined later) in G'. Assign non-zero circulation weights to the edges in G' which are also *isolating weights* for matchings. Then we extract a min-weight generalized perfect matching from G' which in turn extracts a maximum matching from G.

A deterministic construction of non-zero circulation is known in planar bipartite graphs [6], bounded genus bipartite graphs [7] and also in $K_{3,3}$-free and K_5-free bipartite graphs [3]. We construct a graph $G' = (V', E')$ from G by adding vertex t_v with a self loop for each vertex $v \in V$ and join v and t_v using an undirected edge, as shown in Figure 1. Thus, $|V'| = 2n \equiv 0 (\mathrm{mod}\ 2)$. Notice that the genus and the H-minor freeness property of G' remains the same as G. Define a weight function $w' : E' \mapsto \{0, 1\}$ for G' as follows. The original edges of G have weight 1, the self-loops are of weight zero and rest of the new edges have weight 1 (suffices to pick any weight $> 1/2$). We define a *generalized matching* as a set of disjoint edges (possibly) inclusive of self-loops. Various notions for matching naturally extends to generalized matchings. Call a generalized matching as *perfect* wherein every vertex is matched and as *min-weight perfect* if it is perfect and of minimum weight.

▶ **Proposition 7.** *Any matching M in G can be extended to a generalized perfect matching P in G'. Moreover, $w'(P) = n - \nu(G')$.*

Proof. For each $v \in V$ unmatched in G use the (v, t_v) edge of G' in P, thereby matching t_v also. This contributes $(n - 2|M|)$ to $w'(P)$. For the rest of the $2|M|$ vertices $v \in V$ matched in G, match the corresponding new vertices using the self loop at t_v. Since the self loops are of weight zero, the matched edges contribute $|M|$ to $w'(P)$. These form a generalized perfect matching P in G' with $w'(P) = (n - 2|M|) + |M| = n - \nu(G')$. ◀

▶ **Observation 8.** *An extension of a maximum matching in G to G' corresponds to a min-weight generalized perfect matching in G' and a restriction of a min-weight generalized perfect matching of G' to G corresponds to a maximum matching in G.*

Thus the problem of finding a maximum matching in G is equivalent to that of finding a min-weight generalized perfect matching in G'. Now we address the problem of finding isolating weights for extracting a min-weight generalized perfect matching. We define a weight function w (by combining several other weight functions) for which we show the following:

▶ **Lemma 9.** *With respect to the weight function $w : E' \mapsto [\mathbb{N}]$, the min-weight generalized perfect matching in G' is unique.*

To prove this we need some definitions first. Define a *loop-path* as a closed trail $(e_0, e_1, e_2, \ldots, e_k)$ (for k odd, $k > 1$) where e_0 is a self loop, the subtrail $(e_1, e_2, \ldots e_{k-1})$ is a path of non-zero length and e_k is also a self loop. Define a 2-*cycle* as a length 2 directed cycle corresponding to an undirected edge as the underlying graph. Define a 2-*self loop* as a closed walk (e, e) where e is a self loop. Define the alternating weight of a loop-path $\mathcal{P}' = (e_0, e_1, e_2, \ldots, e_k)$ (for $k \geq 2$) to be the alternating sum of the weight of the edges in $\mathcal{P}'$ i.e. $AW(\mathcal{P}') = \sum_{i=0}^{k}(-1)^i w(e_i) = (w(e_0) - w(e_k)) + (-w(e_1) + w(e_2) - \ldots + w(e_{k-1}))$.

Let the graph G' has at most $c'n$ many edges for some constant c'. Define a weight function w'' on the edges of G' which assigns non-zero weights to the self-loops as follows,

$$w''(e) = \left\{ \begin{array}{ll} ic', & \text{if } e = (t_i, t_i) \ 1 \leq i \leq |V| \\ 0 & \qquad\qquad \text{otherwise} \end{array} \right\}$$

The non-zero circulation weights of [3], which works for planar, $K_{3,3}$-free and K_5-free bipartite graphs, compute the weights for the graph directly. For bounded genus graphs the weighting scheme of [7] work on a grid embedding where they use the weighting scheme of [6] to assign the weights. Following [7], given a graph H whose genus is bounded by some constant, the idea is to create a new graph H' with maximum degree 3 by expanding large degree vertices of H into binary trees preserving the bipartition. Now embed H' onto a constant genus grid H'' such that each edge of H' gets expanded into an odd length path in the grid. These are L-reductions preserving the bipartiteness and perfect matchings between H and H'' but *not* maximum matchings. Hence we need to finally *pull back* the weights assigned in H'' to the original graph H ensuring that the non-zero circulation property is preserved.

▶ **Lemma 10.** *The pull-back weights from H'' give non-zero circulation to the cycles in H and are polynomially bounded.*

Denote this non-zero circulation weight for an edge e by $w'''(e)$ which are bounded by, say n^c for some constant c. We combine the weights w'' and w''' into a single weight w^*. Using bit shift, we define the new weight $w^*(e)$ on the edges of G' by $w^*(e) = w''(e) \cdot 2^{\lceil (c+1)\log_2(n) \rceil} + w'''(e)$ for $e \in E(G)$. The weights $w^*(e)$ are bounded by $w''(e) \cdot n^{c+1} \leq c'n \cdot n^{c+1} \leq c'n^{c+2}$. Notice that for the non self-loop edges $w^*(e)$ is bounded by $w'''(e) \leq n^c$.

▶ **Lemma 11.** *With respect to the weighing scheme w^*, the alternating sum of each simple alternating cycle of G' and each loop-path is non-zero.*

Proof. Using the weights $w'''(e)$ from Lemma 10, each simple alternating cycle of G has non-zero circulation, and since each simple cycle of G' is necessarily a simple cycle in G, thus every simple alternating cycle of G' has non-zero circulation. Now consider the loop-path given by $\mathcal{P} = (e_0, e_1, e_2, \dots e_k)$ (for $k \geq 2$). Then, $|AW(\mathcal{P})| = |\sum_{i=0}^{k}(-1)^i w^*(e_k)| \geq |w^*(e_0) - w^*(e_k)| - |(w^*(e_1) - w^*(e_2) + \dots + (-1)^k w^*(e_{k-1}))|$. And,

$$\begin{aligned} |(w^*(e_1) - w^*(e_2) + \dots + (-1)^k w^*(e_{k-1}))| &< |w^*(e_1)| + |w^*(e_2)| + \dots + |w^*(e_{k-1})| \\ &< (k-1)\cdot n^c \quad (\text{as } w^*(e_i) \leq n^c \text{ here}) \\ &< (c'n - 1)\cdot n^c \quad (k < |E(G')| \leq c'n) \\ &\leq c'n^{c+1} \end{aligned}$$

Then $|AW(\mathcal{P})| > |w^*(e_0) - w^*(e_k)| - c'n^{c+1} \geq 0$ and thus every loop-path also has non-zero alternating weight. ◀

Now we combine the weights w' and w^* into a single weight w. Using bit shift again, we define the new weight $w(e)$ on the edges of G' as $w(e) = w'(e) \cdot 2^{\lceil (c+2)\log_2(c'n)\rceil} + w^*(e)$ for $e \in E(G)$. The weights $w(e)$ are bounded by $w'(e) \cdot c'n^{c+2} \leq c'n^{c+2}$ as $w'(e) \in \{0, 1\}$.

▶ **Lemma 12.** *A min-weight generalized perfect matching of G' corresponding to the weight function w' is also a min-weight generalized perfect matching corresponding to the weight function w. Moreover, the alternating sum of the weights with respect to w of simple alternating cycles and loop-paths are non-zero.*

We are now ready to prove Lemma 9.

Proof of Lemma 9. The components of the superposition of any two generalized perfect matchings are either simple alternating cycles, loop-paths, 2-cycles or 2-self-loops. Suppose that there is more than one min-weight generalized perfect matching of G', call them P_1 and P_2, such that $P_1 \neq P_2$. Since $P_1 \neq P_2$, there exists atleast one component of $P_1 \cup P_2$ which is a simple alternating cycle or an loop-path. And since $w''(e)$ assigns a non-zero alternating sum weight on all simple alternating cycle and loop-paths, this implies that the sum of weights of edges from one of P_1 and P_2 is lesser than the other. Swapping the edges between P_1 and P_2 in this component will give rise to a new generalized perfect matching having weight lower than both of P_1 and P_2, which is a contradiction. ◀

We use the determinant polynomial to compute the size of the maximum matching.

▶ **Lemma 13.** *The union of two generalized perfect matchings of G', whose corresponding maximum matchings on G match a distinct set of vertices, do not appear in the determinant polynomial.*

Proof. Since the union of two generalized perfect matchings of G', whose corresponding maximum matchings on G match a distinct set of vertices of G will have an loop-path. Such terms are not represented by any permutation σ, and do not appear in the summation. ◀

Notice that since the generalized perfect matchings containing a loop-path are of larger weight (from the weights w'') than the minimum, we can safely ignore such matchings.

▶ **Observation 14.** *Let P be the unique min-weight generalized perfect matching in G' of weight W. Then the least degree term in the determinant polynomial is x^{2W} corresponding to the unique min-weight cycle cover in G' which is a superposition of P with itself.*

Now we compute the the least degree term in the determiant polynomial using the layered graph method as used in [6]. Start querying from $i = -c'n^{c+2}$ to $+c'n^{c+2}$ to find the first term with non-zero coefficient. Once the weight is known, we can extract P by deleting each edge e in parallel from G' and computing the weight of the min-weight generalized perfect matching in $G' \setminus \{e\}$. If the weight is unchanged, it implies the edge is not in P, and otherwise it is. Once we have P in G', we can find a maximum matching M in G using $M = P \cap E$. Now if $w(P) = W$, then $|M| = n - y$, where $y = \lfloor \frac{W}{2^{\lceil (c+2)\log_2(c'n)\rceil}} \rfloor$. Thus we arrive at the main result of this section:

▶ **Theorem 15.** *Maximum-Matching Search in graphs with non-zero circulation is in SPL.*

Proof. Since the edge weights are polynomially bounded, they can be computed in logspace. Moreover computing the coefficient of a term in the determinant polynomial is in GapL [26]. However since we have used isolating weights, the coefficient of the terms starting from $-c'n^{c+2}$ to the least degree term is either 0 or 1, and hence this computation is in SPL [1]. As a result, we are able to extract a min-weight generalized perfect matching in the graph G' in $\mathsf{L}^{\mathsf{SPL}} = \mathsf{SPL}$. Hence, from the Observation 8 and the previous discussion, we can find a maximum matching of the given graph G in SPL. ◀

4 Reduction from Search to Decision

We reduce the problem of finding a maximum matching to oracle calls for determining the size of a maximum matching in the presence of a parallel algorithm to find a perfect matching and a parallel algorithm to solve the bipartite version of maximum matching.

The reduction uses the classic Gallai-Edmonds theorem (see Theorem 3.2.1 [23]). The crucial observation is based on partition of the vertices given as follows. A vertex $v \in V(G)$ belongs to the set of "deficient" vertices $D(G)$ if there exists some maximum matching of G that leaves v unmatched. A vertex $v \in V(G)$ belongs to $A(G)$, the set of vertices "adjacent" to $D(G)$ if v is a neighbour of some vertex $u \in D(G)$ and $v \notin D(G)$. Rest of the vertices in $V(G) \setminus (D(G) \cup A(G))$ are in the "critical" set $C(G)$. A graph G is said to be *factor-critical* if for every $v \in V(G)$, $\nu(G) = \nu(G - v)$.

▶ **Theorem 16** (Gallai-Edmonds). *Let G be a graph and $D(G), A(G)$, $C(G)$ are defined as above then the components of $D(G)$ are factor-critical and every maximum matching in G*

- *is a perfect matching on $C(G)$,*
- *is near-perfect matching on each component of $D(G)$, and*
- *matches each vertex in $A(G)$ to a distinct component in $D(G)$.*

▶ **Observation 17.** *For a vertex $v \in V(G)$, $v \in D(G)$ if and only if $\nu(G) = \nu(G - v)$.*
Next we can find if $v \in A(G)$ if and only if $v \notin D(G)$ and there is a vertex $u \in D(G)$ such that $v \in N(u)$. Finally, $C(G) = V(G) \setminus (D(G) \cup A(G))$. From the statement of the Gallai-Edmonds theorem it suffices to find:

1. A perfect matching in each connected component of $C(G)$.
2. A maximum matching in the bipartite graph formed by contracting each component of $D(G)$ into a single vertex d and adding an edge to each vertex $a \in A(G)$ such that the corresponding component had an edge to a.
3. A perfect matching in each component of $D(G)$ minus an arbitrary vertex.

▶ **Lemma 18.** *For any class of graphs closed under vertex deletions and edge contractions, there is an NC algorithm for Maximum Matching Search in the class, with oracle queries to Maximum-Matching Size, Maximum-Bipartite-Matching Search and Perfect-Matching Search all for the same class of graphs.*

Then from the recent breakthrough works for NC algorithms for perfect matching respectively in bounded genus [2] and in $K_{3,3}$-free and K_5-free graphs [9] along with our maximum matching algorithm for bipartite graphs from Section 3, we obtain the following,

▶ **Corollary 19.** *Maximum-Matching Search NC-reduces to Maximum-Matching Decision in planar graphs, in bounded genus graphs, in $K_{3,3}$-free graphs and in K_5-free graphs.*

From the above result we also get a pseudo-deterministic NC algorithm for Maximum-Matching Search in the same class of graphs. Recall that pseudo-deterministic algorithms are probabilistic algorithms for search problems that produce a unique output for each given input except with small probability. Since size of the maximum matching can be found in RNC [29] from the above result we get that,

▶ **Theorem 4** (Restated). *Maximum-Matching Search is in pseudo-deterministic NC for planar graphs, bounded genus graphs, $K_{3,3}$-free graphs and K_5-free graphs.*

5 Reducing Count to Bipartite Count

We reduce the problem of counting the number of maximum matchings in a (possibly non-bipartite) graph to oracle calls for counting maximum matchings in bipartite graphs in the presence of a parallel algorithm to count the number of perfect matchings. We do this via a two step process: first reduce the problem of counting maximum matchings in the given graph to counting maximum weight matchings in a bipartite graph and then subsequently reducing the problem of counting of maximum weight matching to counting maximum cardinality matchings while the graph remains bipartite.

This reduction again uses the Gallai-Edmonds decomposition theorem. Recall that, in the decomposition, the vertices in $C(G)$ have a perfect matching and each of the component of $D(G)$ is factor-critical. So we have that the count of maximum matchings in G is the product of the count of the perfect matchings in $C(G)$ and the count of the maximum matchings in $G \setminus C(G) = A(G) \cup D(G)$.

▶ **Lemma 20.** *Maximum-Matching Count L-reduces to weighted Bipartite-Maximum-Matching Count and Maximum-Matching Decision in the presence of Perfect-Matching Count.*

Proof. Let $D_1, D_2, \ldots D_k$ be the components of $D(G)$. Replace each edge (a, d) between a vertex $a \in A(G)$ and a vertex $d \in D_i$, by adding a weight equal to the number of perfect matchings in the component $D_i \setminus d$. Next we contract each component of $D(G)$ into a single vertex d. Replace all the parallel edges between a and d (created due to the contraction) with a single edge (a, d) of weight equal to the sum of weights on the corresponding parallel edges. Since from the Gallai-Edmonds theorem we know that no maximum matching contains an edge between any two vertices of $A(G)$, we have ourselves the weighted bipartite graph instance G'. It is easy to see that the number of maximum matchings in $A(G) \cup D(G)$ equals to the sum $\sum_{M \in \mathcal{MM}_{G'}} wt(M)$ where $\mathcal{MM}_{G'}$ is the set of maximum weighted matchings in G'. This completes the first part of the reduction. In the presence of an oracle access to Perfect-Matching Count the construction is easily seen to be in L. ◀

Gadget Construction. We now replace the weighted bipartite graph G' by an unweighted instance G'' while keeping the counts same. Notice that the the count of the perfect matchings and hence the edge weights takes polynomial (in n) bits, say ℓ bits, to store. For $a \in A(G')$ and $d \in D(G')$, let the weight of the edge (a, d) be $w(a, d)$ and let $b_1 b_2 b_3 \ldots b_\ell$

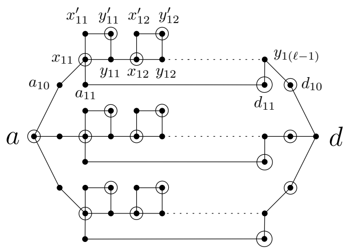

Figure 2 Replacement Gadget $\mathcal{G}_{a,d}$ for each weighted edge (a, d) in G'. The circled and non-circled vertices belong to different bipartitions $A(G'')$ and $D(G'')$ respectively.

be the binary expansion of $w(a, d)$ i.e. $w(a, d) = b_1 2^{\ell-1} + b_2 2^{\ell-2} + \ldots + b_\ell 2^0$. Equivalently, $w(a, d) = \sum_{i \in S} 2^{\ell-i}$ where the set S is the set of indices corresponding to the non-zero bits in the binary expansion of $w(a, d)$. We replace the edge (a, d) by the gadget $\mathcal{G}_{a,d}$ where a and d are connected by $|S|$ many disjoint paths where the i-th path, which corresponds to the bit b_i and is present iff $b_i = 1$ (i.e. these paths are indexed by 1 to ℓ), is of length $2(\ell - i) + 2$. Consider the i-th path. Call the vertices adjacent to a and d as a_{i0} and d_{i0}, respectively. Call the rest of the $2(\ell - i)$ vertices on the path as x_{ij}, y_{ij} alternately for $1 \leq j \leq \ell - i$ where x_{i1} is attached to a_{i0} and $y_{i(\ell-i)}$ is attached to d_{i0}. For the i-th path, add $2(\ell - i)$ new vertices and call them as x'_{ij}, y'_{ij} alternately for $1 \leq j \leq \ell - i$. Add the undirected edges $(x_{ij}, x'_{ij}), (x'_{ij}, y'_{ij}), (y_{ij}, y'_{ij})$ for all $i \in S$ and $1 \leq j \leq \ell - i$. Each such modified path is informally called as *box-path*. Connect x_{i1} and $y_{i(\ell-i)}$ with a separate path $x_{i1}, a_{i1}, d_{i1}, y_{i(\ell-i)}$ of length 3 where each consecutive vertex has an edge between them. See Figure 2 where we assume $b_1 = 1$. Notice that the graph remains bipartite after attaching the gadgets.

▶ **Lemma 21.** *The gadget $\mathcal{G}_{a,d}$ has the following properties:*

1. *There is a unique perfect matchings in $\mathcal{G}_{a,d} \setminus \{a, d\}$.*
2. *If a is matched inside $\mathcal{G}_{a,d}$ then $\mathcal{G}_{a,d}$ has a perfect matching. If a is matched with a vertex outside $\mathcal{G}_{a,d}$, then $\mathcal{G}_{a,d} \setminus \{a\}$ has a near-perfect matching.*
3. *The vertices a and d remain in different bipartitions. A vertex $v \in \mathcal{G}_{a,d}$ is in $A(G'')$ if only if it is in the same bipartition as a. Rest of the vertices are in $D(G'')$.*
4. *There are $3w(a, d)$ many maximum (either perfect or near-perfect) matchings in $\mathcal{G}_{a,d}$.*

Recall that, in a maximum matching in the weighted graph G' all the $A(G')$ edges are matched. Since for each edge (a, d) in the matching we get a factor 3 extra in the count than the desired $w(a, d)$ (notice that these weights are multiplicative), we finally divide the count of the maximum matchings in the bipartite unweighted graph G'' by $3^{(|A(G')|)}$ to get the correct count. For any $w \in \mathbb{N}$ we can construct such a gadget and replace every non-unit weight edge of G' by the gadget of the corresponding weight. This completes the second part of the reduction. Since other than counting perfect matchings all the steps of the reduction can be done in L, we have that:

▶ **Theorem 22.** *Maximum-Matching Count* L*-reduces to Bipartite-Maximum-Matching Count and Maximum-Matching Decision in the presence of Perfect-Matching Count.*

A modification of the result of [11] combined with the techniques from [24] gives an NC algorithm for counting perfect matchings in logarithmic genus graphs [25]. Vazirani [36] and Straub et al. [33] show that counting perfect matchings is in NC in $K_{3,3}$-free graphs and K_5-free graphs respectively. And hence, along with Theorem 22 we have the following result,

▶ **Corollary 23.** *Maximum-Matching Count* NC*-reduces to Bipartite-Maximum-Matching Count and Maximum-Matching Decision in planar, bounded genus,* $K_{3,3}$*-free and* K_5*-free graphs.*

6 Conclusion and Open Ends

The main contribution of this investigation is a better complexity bound on bipartite planar maximum matching which matches the upper bound for bipartite planar perfect matching. We also show an NC reduction from planar maximum matching search to planar maximum matching decision along with an NC reduction from counting planar maximum matchings to counting bipartite planar maximum matchings and planar maximum matching decision (where the NC-bounds hide the complexity of finding and counting planar perfect matching, respectively). To reiterate, the main open questions are to find NC algorithms to determine the size of planar maximum matching and for counting bipartite planar maximum matchings.

References

1. E. Allender, K. Reinhardt, and S. Zhou. Isolation, Matching, and Counting: Uniform and Nonuniform Upper Bounds. *Journal of Computer and System Sciences*, 59:164–181, 1999.
2. Nima Anari and Vijay V. Vazirani. Planar Graph Perfect Matching is in NC. *CoRR*, abs/1709.07822, 2017.
3. Rahul Arora, Ashu Gupta, Rohit Gurjar, and Raghunath Tewari. Derandomizing Isolation Lemma for K3,3-free and K5-free Bipartite Graphs. In *33rd Symposium on Theoretical Aspects of Computer Science, STACS*, pages 10:1–10:15, 2016.
4. Samir Datta, Raghav Kulkarni, Nutan Limaye, and Meena Mahajan. Planarity, Determinants, Permanents, and (Unique) Matchings. *ACM Trans. Comput. Theory*, 1(3):1–20, 2010.
5. Samir Datta, Raghav Kulkarni, and Anish Mukherjee. Space-Efficient Approximation Scheme for Maximum Matching in Sparse Graphs. In *41st International Symposium on Mathematical Foundations of Computer Science, MFCS*, pages 28:1–28:12, 2016.
6. Samir Datta, Raghav Kulkarni, and Sambuddha Roy. Deterministically Isolating a Perfect Matching in Bipartite Planar Graphs. *Theory of Computing Systems*, 47:737–757, 2010.
7. Samir Datta, Raghav Kulkarni, Raghunath Tewari, and N. V. Vinodchandran. Space complexity of perfect matching in bounded genus bipartite graphs. *J. Comput. Syst. Sci.*, 78(3):765–779, 2012.
8. J. Edmonds. Paths, Trees and Flowers. *Canad. J. Math.*, 17:449–467, 1965.
9. David Eppstein and Vijay V. Vazirani. NC algorithms for perfect matching and maximum flow in one-crossing-minor-free graphs. *CoRR*, abs/1802.00084, 2018.
10. Stephen A. Fenner, Rohit Gurjar, and Thomas Thierauf. Bipartite perfect matching is in quasi-NC. In *48th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 754–763, 2016.
11. Anna Galluccio and Martin Loebl. On the Theory of Pfaffian Orientations. I. Perfect Matchings and Permanents. *Electr. J. Comb.*, 6, 1999.
12. Eran Gat and Shafi Goldwasser. Probabilistic Search Algorithms with Unique Answers and Their Cryptographic Applications. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:136, 2011.
13. Oded Goldreich, Shafi Goldwasser, and Dana Ron. On the possibilities and limitations of pseudodeterministic algorithms. In *Innovations in Theoretical Computer Science, ITCS*, pages 127–138, 2013.

14 Shafi Goldwasser and Ofer Grossman. Bipartite Perfect Matching in Pseudo-Deterministic NC. In *44th International Colloquium on Automata, Languages, and Programming, ICALP*, pages 87:1–87:13, 2017.

15 Shafi Goldwasser, Ofer Grossman, and Dhiraj Holden. Pseudo-Deterministic Proofs. In *9th Innovations in Theoretical Computer Science Conference, ITCS*, pages 17:1–17:18, 2018.

16 Ofer Grossman. Finding Primitive Roots Pseudo-Deterministically. *Electronic Colloquium on Computational Complexity (ECCC)*, page 207, 2015.

17 Ofer Grossman and Yang P. Liu. Reproducibility and Pseudo-Determinism in Log-Space. *CoRR*, abs/1803.04025, 2018.

18 Thanh Minh Hoang. On the Matching Problem for Special Graph Classes. In *IEEE Conference on Computational Complexity*, pages 139–150, 2010. `doi:10.1109/CCC.2010.21`.

19 Stefan Hougardy and Doratha E. Drake Vinkemeier. Approximating weighted matchings in parallel. *Inf. Process. Lett.*, 99(3):119–123, 2006.

20 Marek Karpinski and Wojciech Rytter. *Fast parallel algorithms for graph matching problems*, volume 9 of *Oxford Lecture Series in Mathematics and its Applications.* The Clarendon Press, Oxford University Press, New York, 1998.

21 P W Kastelyn. Graph Theory and Crystal Physics. In F Harary, editor, *Graph Theory and Theoretical Physics*, pages 43–110. Academic Press, 1967.

22 Raghav Kulkarni, Meena Mahajan, and Kasturi R. Varadarajan. Some perfect matchings and perfect half-integral matchings in NC. *Chicago Journal of Theoretical Computer Science*, 2008(4), September 2008.

23 L. Lovász and M.D. Plummer. *Matching Theory*, volume 29. North-Holland Publishing Co, 1986.

24 Meena Mahajan, P. R. Subramanya, and V. Vinay. The combinatorial approach yields an NC algorithm for computing Pfaffians. *Discrete Applied Mathematics*, 143(1-3):1–16, 2004.

25 Meena Mahajan and Kasturi R. Varadarajan. A new NC-algorithm for finding a perfect matching in bipartite planar and small genus graphs. In *STOC*, pages 351–357, 2000.

26 Meena Mahajan and V. Vinay. Determinant: Combinatorics, Algorithms, and Complexity. *Chicago J. Theor. Comput. Sci.*, 1997.

27 Silvio Micali and Vijay V. Vazirani. An O(sqrt(|v|) |E|) Algorithm for Finding Maximum Matching in General Graphs. In *21st Annual Symposium on Foundations of Computer Science*, pages 17–27, 1980.

28 Gary L. Miller and Joseph Naor. Flow in Planar Graphs with Multiple Sources and Sinks. *SIAM J. Comput.*, 24(5):1002–1017, 1995.

29 Ketan Mulmuley, Umesh Vazirani, and Vijay Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7:105–113, 1987.

30 Jaroslav Nesetril and Patrice Ossona de Mendez. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics.* Springer, 2012.

31 Igor Carboni Oliveira and Rahul Santhanam. Pseudodeterministic constructions in subexponential time. In *49th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 665–677, 2017.

32 Piotr Sankowski. NC algorithms for weighted planar perfect matching and related problems. In *45th International Colloquium on Automata, Languages, and Programming, ICALP*, pages 97:1–97:16, 2018.

33 Simon Straub, Thomas Thierauf, and Fabian Wagner. Counting the Number of Perfect Matchings in K 5-Free Graphs. *Theory Comput. Syst.*, 59(3):416–439, 2016.

34 Ola Svensson and Jakub Tarnawski. The Matching Problem in General Graphs Is in Quasi-NC. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 696–707, 2017.

35 L. G. Valiant. The Complexity of Computing the Permanent. *Theor. Comput. Sci.*, 8:189–201, 1979.

36 Vijay Vazirani. NC algorithms for computing the number of perfect matchings in $k_{3,3}$–free graphs and related problems. In *Proceedings of SWAT '88*, pages 233–242, 1988.

37 Douglas B. West. *Introduction to Graph Theory*. Prentice Hall, 2 edition, September 2000.