Bipartite Perfect Matching is in quasi-NC

Jędrzej Kula

27 stycznia 2022

THEORETICAL COMPUTER SCIENCE
Jagiellonian University

Table of contents

Definition

Uniform circuit *- The circuit where local queries about the circuit can be answered in poly-logarithmic time.*

Definition

NC^k *- is the class of decision problems decidable by uniform boolean circuits with a polynomial number of gates of at most two inputs and depth* $O(log^i n)$*, or the class of decision problems solvable in time* $O(log^i n)$ *on a parallel computer with a polynomial number of processors.*

Definition

NC *- the union of classes* NC^k*, over all* $k \geq 0$*.*

Definition

quasi-NC^k - is the class of decision problems decidable by uniform boolean circuits with a quasi-polynomial $2^{log^{O(1)}n}$ number of gates of at most two inputs and depth $O(log^k n)$.

Definition

quasi-NC *- the union of classes $quasi - NC^k$, over all $k \geq 0$.*

Definition

Isolating weight function *- function w is called isolating for G, if there is a unique perfect matching of minimum weight in G.*

Lemma

Lemma 1.1 *(Isolation Lemma) For a graph $G(V, E)$, let w be a random weight assignment, where edges are assigned weights chosen uniformly and independently at random from $\{1, 2, \dots, 2|E|\}$. Then w is isolating with probability $\geq \frac{1}{2}$.*

An RNC algorithm for Search-PM

Let L and R be vertex partitions of G, let w be a weight function of G. Consider the following $\frac{n}{2} \times \frac{n}{2}$ matrix A associated with G,

$$A(i,j) = \begin{cases} 2^{w(e)}, \text{ if } e = (l_i, r_j) \in E, \\ 0, \text{ otherwise.} \end{cases}$$

The algorithm for SEARCH-PM computes the determinant of A. This determinant is the signed sum over all perfect matchings in G:

$$det(A) = \sum_{\pi \in S_{\frac{n}{2}}} sgn(\pi) \prod_{i=1}^{\frac{n}{2}} A(i, \pi(i))$$

$$= \sum_{M \ pm \ in \ G} sgn(M) 2^{w(M)}$$

If G does not have a perfect matching, then $det(A) = 0$. But such result can be also an outcome of cancellations due to $sgn(M)$. To avoid this situation, w needs to be designed correctly. In particular if G has a perfect matching and w is isolating, then $det(a) \neq 0$ (since the term corresponding to the minimum weight perfect matching cannot be canceled).

Given an insolating weight assignment for G, it is possible to construct the minimum wieght perfect matching in NC.

Let M^* be the unique minimum weight perfect matching in G.
$w(M^*)$ is equal to the highest power of 2 dividing $det(A)$.
For every edge $e \in E$ we can compute $det(A_e)$, where A_e is matrix associated with $G - e$. If the highest power of 2 that divides $det(A_e)$ is larget then $2^{w(M^*)}$, then $e \in M^*$. It can be done in parallel to find all edges of M^*.

The Matching Polytope

Definition

The perfect matching point of a graph G *is a point in the edge space(*$\mathbb{R}^{|E|}$*). For any perfect matching* M *of* G*, consider its incidence vector* $x^M = (x_e^M)_e \in \mathbb{R}^{|E|}$ *given by*

$$x_e^M = \begin{cases} 1, & \textit{if } e \in M, \\ 0, & \textit{otherwise.} \end{cases}$$

Definition

The perfect matching polytope $PM(G)$ *of a graph* G *is a polytope in the edge space(*$\mathbb{R}^{|E|}$*). It is defined to be the convex hull of all its perfect matching points.*

For $w\colon E \to R$ and $x = (x_e)_e \in \mathbb{R}^{|E|}$:

$$w(x) = \sum_{e \in E} w(e) x_e$$

$$w(M) = w(x^M)$$

Lemma

Lemma 2.1 *Let G be a bipartite graph and $x = (x_e)_e \in \mathbb{R}^{|E|}$. Then $x \in PM(G)$ if and only if*

$$\sum_{e \in \delta(e)} x_e = 1 \ v \in V$$

and

$$x_e \geq 0 \ e \in E$$

where $\delta(v)$ denotes the set of edges incident on the vertex v.

For general graphs, the polytope described by such conditions can have vertices which are not perfect matchings.

Nice Cycles and Circulation

Definition

A cycle C in G is nice, if the subgraph $G - C$ still has a perfect matching. In other words, it can be obtained from the symmetric difference of two perfect matching. It is always an even cycle.

Definition

The circulation $c_w(C)$ of an even cycle C is the alternating sum of the edge weights of C,

$$c_w(C) = |w(v_1, v_2) - w(v_2, v_3) + w(v_3, v_4) - \cdots - w(v_k, v_1)|$$

Lemma

Lemma 2.2 *Let G be a graph with a perfect matching, and let w be a weight function that all nice cycles in G have nonzero circulation. The the minimum perfect matching is unique. That is, w is isolating.*

Lemma

Lemma 2.3 *Let G be a graph with n nodes. Then, for any number s, one can construct a set of $O(n^2s)$ weight assignments with weights bounded by $O(n^2s)$, such that for any set of s cycles, one of the assignments gives nonzero circulation to each of the s cycles.*

Table of contents

Main ideas

- For any two $M \in PM(G)$ the edges where they differ form disjoint cycles.
- For a cycle C, $c_w(C)$ is defined to be the difference of weights of two perfect matchings which differ exactly on the edges of C.
- Lemma 2.2, but it is not clear if there exists such a wieght assignment with small weights.
- We use a weight function that has nonzero circulations only for small cycles.

Main ideas

- We consider the subgraph G', which is the union of minimum weight perfect matching in G.
- In bipartite case it is not only smaller, but also does not contain any small cycles.
- We show that if graph has no cycles of length $< r$, then the number of cycles of length $< 2r$ is polynomially bounded.
- For $\log n$ rounds: in the i-th round, assign weight which ensure nonzero circulations for all cycles with length $< 2^i$. Since the graph obtained after $(i-1)$-th rounds has no cycles of lenth $< 2^{i-1}$, the number of cycles of length $< 2^i$ is small.
- In $\log n$ rounds, we get a unique minimum weight perfect matching.

The union of Minimum Weight Perfect Matchings

Lemma

Lemma 3.2 *Let $G(V, E)$ be a bipartite graph with weight function w. Let C be a cycle in G such that $c_w(C) \neq 0$. Let E_1 be the union of all minimum weight perfect matchings in G. Then graph $G_1(V, E_1)$ does not contain cycle C.*

Proof. Let the weight of the minimum weight perfect matchings in G be q. Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t$ be all the minimum weight perfect matching points of G, i.e., the corners of $\mathrm{PM}(G)$ corresponding to the weight q. Consider the average point $\boldsymbol{x} \in \mathrm{PM}(G)$ of these matching points,

$$\boldsymbol{x} = \frac{\boldsymbol{x}_1 + \boldsymbol{x}_2 + \cdots + \boldsymbol{x}_t}{t}.$$

Clearly, $w(\boldsymbol{x}) = q$. Since each edge in E_1 participates in a minimum weight perfect matching, for $\boldsymbol{x} = (x_e)_e$, we have that $x_e \neq 0$ for all $e \in E_1$. Now, consider a cycle C with $c_w(C) \neq 0$. Let the edges of cycle C be $(e_1, e_2, \ldots, e_p)$ in cyclic order. For the sake of contradiction let us assume that all the edges of C lie in E_1. We show that when we move from point $\boldsymbol{x}$ along the cycle C, we reach a point in the perfect matching polytope with a weight smaller than q. This technique of moving along the cycle has been used by Mahajan and Varadarajan [MV00]. To elaborate, consider a new point $\boldsymbol{y} = (y_e)_e$ such that for all $e \in E$,

$$y_e = \begin{cases} x_e + (-1)^i \, \varepsilon, & \text{if } e = e_i, \text{ for some } 1 \leq i \leq p, \\ x_e, & \text{otherwise,} \end{cases}$$

for some $\varepsilon \neq 0$. Clearly, the vector $\boldsymbol{x} - \boldsymbol{y}$ has nonzero coordinates only on cycle C, where its entries are alternating ε and $-\varepsilon$. Hence,

$$w(\boldsymbol{x} - \boldsymbol{y}) = \pm \varepsilon \cdot c_w(C). \tag{5}$$

As $c_w(C) \neq 0$, we get $w(\boldsymbol{x} - \boldsymbol{y}) = w(\boldsymbol{x}) - w(\boldsymbol{y}) \neq 0$. We choose $\varepsilon \neq 0$ such that

- its sign is such that $w(\boldsymbol{y}) < w(\boldsymbol{x}) = q$, and

We argue that $\boldsymbol{y}$ fulfills the conditions of Lemma 2.1 and therefore also lies in the perfect matching polytope. Because $y_e \geq 0$ for all $e \in E$, it satisfies inequality (4) from Lemma 2.1. It remains to show that $\boldsymbol{y}$ also satisfies

$$\sum_{e \in \delta(v)} y_e = 1 \qquad v \in V. \tag{6}$$

To see this, let $v \in V$. We consider two cases:

1. $v \notin C$. Then $y_e = x_e$ for each edge $e \in \delta(v)$. Thus, we get (6) from equation (3) for $\boldsymbol{x}$.
2. $v \in C$. Let e_j and e_{j+1} be the two edges from C which are incident on v. By definition, $y_{e_j} = x_{e_j} + (-1)^j \varepsilon$ and $y_{e_{j+1}} = x_{e_{j+1}} + (-1)^{j+1} \varepsilon$. For any other edge $e \in \delta(v)$, we have $y_e = x_e$. Combining this with equation (3) for $\boldsymbol{x}$, we get that $\boldsymbol{y}$ satisfies (6) for v.

We conclude that $\boldsymbol{y}$ lies in the polytope $\mathrm{PM}(G)$. Since $w(\boldsymbol{y}) < q$, there must be a corner point of the polytope, which corresponds to a perfect matching in G with weight $< q$. This gives a contradiction. $\square$

Corollary

Corollary 3.3 *Let $G(V, E)$ be a bipartite graph with weight function w. Let E_1 be the union of all minimum weight perfect matchings in G. Then every perfect matching in the graph $G_1(V, E_1)$ has the same weight - the minimum weight of any perfect matching in G.*

Lemma

Lemma 3.4 *Let H be a graph with n nodes that has no cycles of length $\leq r$. Let $r' = 2r$ when r is even, and $r' = 2r - 2$ otherwise. Then H has $\leq n^4$ cycles of length $\leq r'$.*

Proof. Let $C = (v_0, v_1, \ldots, v_{\ell-1})$ be a cycle of length $\ell \leq r'$ in G. Let $f = \ell/4$. We successively choose four nodes on C with distance $\leq \lceil f \rceil \leq r/2$ and *associate* them with C. We start with $u_0 = v_0$ and define $u_i = v_{\lceil if \rceil}$, for $i = 1, 2, 3$. Note that the distance between u_3 and u_0 is also $\leq \lceil f \rceil$. Since we could choose any node of C as starting point u_0, the four nodes (u_0, u_1, u_2, u_3) associated with C are not uniquely defined. However, they uniquely describe C.

Claim 1. *Cycle C is the only cycle in H of length $\leq r'$ that is associated with* (u_0, u_1, u_2, u_3).

Proof. Suppose $C' \neq C$ would be another such cycle. Let $p \neq p'$ be paths of C and C', respectively, that connect the same u-nodes. Note that p and p' create a cycle in H of length at most

$$|p| + |p'| \;\leq\; \frac{r}{2} + \frac{r}{2} \;\leq\; r,$$

which is a contradiction. This proves the claim. □

There are $\leq n^4$ ways to choose 4 nodes and their order. By Claim 1, this gives a bound on the number of cycles of length $\leq r'$. □

Constructing the Weight Assignment

Let $G(V, E) = G_0$ be bipartite graph with n nodes that has perfect matchings. Define $k = \log n - 1$. Note that the shortest cycles have length 4. Define

w_i: a weight function such that all cycles in G_i of length $\leq 2^{i+2}$ have nonzero circulations.

G_{i+1}: the union of minimum weight perfect matchings in G_i according to weight w_i.

By the definition of G_i, any two perfect matchings in G_i have the same weight, not only according to w_i, but also to w_j for all $j < i$, for any $1 \leq i \leq k$.
By Lemma 3.2, graph G_i does not have any cycles of length $\leq 2^{i+1}$ for each $1 \leq i \leq k$. In particular, G_k does not have any cycles, since $2^{k+1} \geq n$. Therefore G_k has a unique perfect matching.

Final weight function w will be a combination of $w_0, w_1, \dots, w_{k-1}$. We combine them in a way that the weight assignment in a later round does not interfere with the order of perfect matchings given by earlier round weights. Let B be a number greater than the weight of any edge under any of these weight assignments. Then, define

$$w = w_0 B^{k-1} + w_1 B^{k-2} + \cdots + w_{k-1} B^0.$$

In the definition of w, the precedence decreases from w_0 to w_{k-1}.

For any two perfect matchings M_1 and M_2 in G_0, we have $w(M_1) < w(M_2)$, if and only if there exists an $0 \leq i \leq k-1$ such that

$$w_j(M_1) = w_j(M_2), \text{ j < i},$$

$$w_i(M_1) < w_i(M_2).$$

The perfect matchings left in G_i have a strictly smaller weight with respect to w than the ones in G_{i-1} that did not make G_i.

Lemma

Lemma 3.5. *For any $1 \leq i \leq k$, let M_1 be a perfect matching in G_i and M_2 be a perfect matching in G_{i-1} which is not in G_i. Then $w(M_1) < w(M_2)$.*

Proof. Since M_1 and M_2 are perfect matching in G_{i-1}, we have $w_j(M_1) = w_j(M_2)$, for all $j < i - 1$. From the definition of G_i and Corollary 3.3, it follows that $w_{i-1}(M_1) < w_{i-1}(M_2)$. Hence we get that $w(M_1) < w(M_2)$.

It follows that the unique perfect matching in G_k has a strictly smaller weight with respect to w than all other perfect matchings.

Corollary

Corollary 3.6. *The weight assignment*

$$w = w_0 B^{k-1} + w_1 B^{k-2} + \cdots + w_{k-1} B^0$$

is isolating for G_0.

It remains to bound the values of the weights assigned. In the first round we give nonzero circulation to all cycles of length 4. The number of such cycles is $\leq n^4$. In the i-th round, we have graph G_i that does not have any cycles of length $\leq 2^{i+1}$. For G_i, we give nonzero circulation to all cycles of length $\leq 2^{i+2}$. By Lemma 3.4, the number of sych cycles is $\leq n^4$. Therefore, each w_i needs to give nonzero circulations to $\leq n^4$ cycles, for $0 \leq i < k$.

Now we apply Lemma 2.3 with $s = n^4$. This yields a set of $O(n^6)$ weight assignments with weights bounded by $O(n^6)$. Recall that the number B used in definition of w is the highest weight assigned by any w_i, so $B = O(n^6)$. Therefore the weights in the assignment w are bounded by $B^k = O(n^{6logn})$. That is, the weights have $O(\log^2 n)$ bits.

For each w_i we have $O(n^6)$ possibilities and we need to try all of them. In total, we need to try $O(n^{6k}) = O(n^{6 \log n})$ weight assignments in parallel. Every weight assignment can be constructed in quasi-NC^1 with circut size $2^{O(\log^2 n)}$.

Lemma

Lemma 3.7. *In quasi-NC^1, one can construct a set of $O(n^{6 \log n})$ integer weight functions on $[\frac{n}{2}] \times [\frac{n}{2}]$, where the weights have $O(\log^2 n)$ bits, such that for any given bipartite graph with n nodes, one of the weight functions is isolating.*

With this construction of weight functions, we can decide the existence of a perfect matching in a bipartite graph in quasi-NC^2 as follows:

- Recall the bi-adjacency matrix A which has entry $2^w(e)$ for edge e.
- Compute $det(A)$ for each of the constructed weight functions in parallel.
- If the given graph has a perfect matching, then one of the weight funtions isolates a perfect matching (for this $det(A)$ will be nonzero).
- When there is no perfect matching, then $det(A)$ will be zero for any weight funtion.

Weights constructed in this way have $O(\log^2 n)$ bits, so the determinant entries have quasi-polynomial bits. The determinant can be computed in parallel, with circuits of quasi-polynomial size $2^{O(\log^2 n)}$. We need to compute $2^{O(\log^2 n)}$-many determinants in parallel, so the algorithm is in quasi-NC^2 with circuit size $2^{O(\log^2 n)}$.

To construct a perfect matching we want to follow algorithm presented at the beginning with each of weight functions.
For a weight funtion w which is isolating, the algorithm outputs the unique minimum weight perfect matching M. If we have a weight funtion w' which is not isolating, still $det(A)$ might be non-zero with respect to w'. Then the algorithm computes a set of edges M' that might or might not be a perfect matching. We can verify if M' is perfect matching, and in this case, we will output M'. As the algorithm involves computation of similar determinants as in the decision algorithm, it is in quasi-NC^2 with circut size $2^{O(\log^2 n)}$.

Table of contents

Decision Version

Theorem 4.1. *For bipartite graphs, there is an* RNC^2*-algorithm for* PM *which uses* $O(\log^2 n)$ *random bits.*

To prove Theorem 4.1, consider our algorithm from Section 3. There are two reasons that we need quasi-polynomially large circuits: (i) we need to try quasi-polynomially many different weight assignments and (ii) each weight assignment has quasi-polynomially large weights. We show how to come down to polynomial bounds in both cases by using randomization.

To solve the first problem, we modify Lemma 2.3 to get a random weight assignment which works with high probability.

Lemma 4.2 ([CRS95, KS01]). *Let G be a graph with n nodes and $s \geq 1$. There is a random weight assignment w which uses $O(\log ns)$ random bits and assigns weights bounded by $O(n^3 s \log ns)$, i.e., with $O(\log ns)$ bits, such that for any set of s cycles, w gives nonzero circulation to each of the s cycles with probability at least $1 - 1/n$.*

Proof. We follow the construction of Lemma 2.3 and give exponential weights modulo small numbers. Here, we use only prime numbers as moduli. Recall the weight function w defined by $w(e_i) = 2^{i-1}$. Let us choose a random number p among the first t prime numbers. We take our random weight assignment to be $w \bmod p$. We want to show that with high probability this weight function gives nonzero circulation to every cycle in $\{C_1, C_2, \ldots, C_s\}$. In other words, $\prod_{i=1}^{s} c_w(C_i) \not\equiv 0 \pmod{p}$. As the product is bounded by $2^{n^2 s}$, it has at most $n^2 s$ prime factors. Let us choose $t = n^3 s$. This would mean that a random prime works with probability at least $(1 - 1/n)$. As the t-th prime can only be as large as $2t \log t$, the weights are bounded by $2t \log t = O(n^3 s \log ns)$, and hence have $O(\log ns)$ bits. A random prime with $O(\log ns)$ bits can be constructed using $O(\log ns)$ random bits (see [KS01]). ☐

Recall from Section 3.2 that for a bipartite graph G with n nodes, we had $k = \lceil \log n \rceil - 1$ rounds and constructed one weight function in each round. We do the same here, however, we use the random scheme from Lemma 4.2 to choose each of the weight functions $w_0, w_1, \ldots, w_{k-1}$ independently. The probability that all of them provide nonzero circulation on their respective set of cycles $\geq 1 - k/n \geq 1 - \log n/n$ using the union bound.

Now, instead of combining them to form a single weight assignment, we use a different variable for each weight assignment. We modify the construction of matrix A from Section 2.2.

Let $L = \{u_1, u_2, \ldots, u_{n/2}\}$ and $R = \{v_1, v_2, \ldots, v_{n/2}\}$ be the vertex partition of G. For variables $x_0, x_1, \ldots, x_{k-1}$, define an $n/2 \times n/2$ matrix A by

$$A(i,j) = \begin{cases} x_0^{w_0(e)} x_1^{w_1(e)} \cdots x_{k-1}^{w_{k-1}(e)}, & \text{if } e = (u_i, v_j) \in E, \\ 0, & \text{otherwise.} \end{cases}$$

From arguments similar to those in Section 2.2, one can write

$$\det(A) = \sum_{M \text{ perfect matching in } G} \operatorname{sgn}(M)\, x_0^{w_0(M)} x_1^{w_1(M)} \cdots x_{k-1}^{w_{k-1}(M)},$$

where $\operatorname{sgn}(M)$ is the sign of the corresponding permutation. From the construction of the weight assignments it follows that if the graph has a perfect matching then the lexicographically minimum term in $\det(A)$, with respect to the exponents of variables $x_0, x_1, \ldots, x_{k-1}$

in this precedence order, comes from a unique perfect matching. Thus, we get the following lemma.

Lemma 4.3. $\det(A) \neq 0 \iff$ *G has a perfect matching.*

Recall that each w_i needs to give nonzero circulations to n^4 cycles. Thus, the weights obtained by the scheme of Lemma 4.2 will be bounded by $O(n^7 \log n)$. This means the weight of a matching will be bounded by $O(n^8 \log n)$. Hence $\det(A)$ is a polynomial of individual degree $O(n^8 \log n)$ with $\log n$ variables. To test if $\det(A)$ is nonzero one can apply the standard randomized polynomial identity test [Sch80, Zip79, DL78]. That is, to plug in random values for variables x_i, independently from $\{1, 2, \dots, n^9\}$. If $\det(A) \neq 0$, then the evaluation is nonzero with high probability.

Number of random bits: For a weight assignment w_i, we need $O(\log ns)$ random bits from Lemma 4.2, where $s = n^4$. Thus, the number of random bits required for all w_i's together is $O(k \log n) = O(\log^2 n)$. Finally, we need to plug in $O(\log n)$ random bits for each x_i. This again requires $O(\log^2 n)$ random bits.

Complexity: The weight construction involves taking exponential weights modulo small primes by Lemma 4.2. Primality testing can be done by the brute force algorithm in NC^2, as the numbers involved have $O(\log n)$ bits. Thus, the weight assignments can be constructed in NC^2. Moreover, the determinant with polynomially bounded entries can be computed in NC^2 [Ber84].

In summary, we get an RNC^2-algorithm that uses $O(\log^2 n)$ random bits as claimed in Theorem 4.1.

Search Version

Theorem 4.4. *For bipartite graphs, there is an* RNC^3*-algorithm for* Search-PM *which uses* $O(\log^2 n)$ *random bits.*

Let again $G(V, E)$ be the given bipartite graph with vertex partition $L = \{u_1, u_2, \ldots, u_{n/2}\}$ and $R = \{v_1, v_2, \ldots, v_{n/2}\}$. We construct the weight assignments $w_0, w_1, \ldots, w_{k-1}$ as in Lemma 4.2 in the randomized decision version. Let M^* be the unique minimum weight perfect matching in G with respect to the combined weight function w. Let $w_r(M^*) = w_r^*$, for $0 \leq r < k$.

Recall from Section 3.2 the sequence of subgraphs $G_1, G_2, \ldots, G_k$ of $G = G_0$, where G_{r+1} consists of the minimum perfect matchings of G_r according to weight w_r. In order to compute M^*, we would like to actually construct all the graphs $G_1, G_2, \ldots, G_k$. However, it is not clear how to achieve this with $O(\log^2 n)$ random bits. Instead, we will construct a sequence of graphs $H_1, H_2, \ldots, H_k$ such that H_r will be a subgraph of G_r, for each $1 \leq r \leq k$. Furthermore, each H_r will contain the matching M^*. Recall that G_k consists of the unique perfect matching M^*. Hence, once we have $H_k = G_k$, we are done.

Let $H_0 = G$ and $0 \le r < k$. We describe the r-th round. Suppose we have constructed the graph $H_r(V, E_r)$ and want to compute H_{r+1}. An edge will appear in H_{r+1} only if it participates in a matching M with $w_r(M) = w_r^*$. Thus, we will have that H_{r+1} is a subgraph of G_{r+1}. For an edge e, let $\boldsymbol{X}_r^{\boldsymbol{w}(e)}$ denote the product

$$\boldsymbol{X}_r^{\boldsymbol{w}(e)} = x_r^{w_r(e)} x_{r+1}^{w_{r+1}(e)} \cdots x_{k-1}^{w_{k-1}(e)} .$$

For a matching M, the term $\boldsymbol{X}_r^{\boldsymbol{w}(M)}$ is defined similarly. Let $N(e)$ denote the set of edges which are neighbors of an edge e in G_r, i.e. all edges $e' \neq e$ that share an endpoint with e. For an edge $e \in E_r$, define the $n/2 \times n/2$ matrix A_e as

$$A_e(i,j) = \begin{cases} \boldsymbol{X}_r^{\boldsymbol{w}(e')}, & \text{if } e' = (u_i, v_j) \in E_r - N(e), \\ 0, & \text{otherwise.} \end{cases}$$

Note that the matrix A_e has a zero entry for each neighboring edge of e. Thus, its determinant is a sum over all perfect matchings which contain e. That is,

$$\det(A_e) = \sum_{\substack{M \text{ pm in } H_r \\ e \in M}} \operatorname{sgn}(M)\, \boldsymbol{X}_r^{\boldsymbol{w}(M)} .$$

Consider the coefficient c_e of $x_r^{w_r^*}$ in $\det(A_e)$,

$$c_e = \sum_{\substack{M \text{ pm in } H_r \\ w_r(M) = w_r^*,\, e \in M}} \operatorname{sgn}(M)\, \boldsymbol{X}_{r+1}^{\boldsymbol{w}(M)} .$$

Define the graph H_{r+1} to be the union of all the edges e for which the polynomial $c_e \neq 0$. We claim that each edge of M^* appears in H_{r+1}. For any edge $e \in M^*$, the polynomial c_e will contain the term $\boldsymbol{X}_{r+1}^{\boldsymbol{w}(M^*)}$. As the matching M^* is isolated in H_r with respect to the weight vector $(w_{r+1}, \ldots, w_{k-1})$, the polynomial c_e is nonzero.

For the construction of H_{r+1}, we need to test if c_e is nonzero, for each edge e in H_r. As argued above in the decision part, the degree of c_e is $O(n^8 \log^2 n)$. We apply the standard zero-test, i.e., we plug in random values for the variables $x_{r+1}, \ldots, x_{k-1}$ independently from $\{1, 2, \ldots, n^{11}\}$. The probability that the evaluation will be nonzero is at least $1 - O(\log^2 n/n^3)$. To compute this evaluation, we plug in values of $x_{r+1}, \ldots, x_{k-1}$ in $\det(A_e)$ and find the coefficient of $x_r^{w_r^*}$. This can be done in NC^2 [BCP84, Corollary 4.4]. For all the edges, we use the same random values for variables $x_{r+1}, \ldots, x_{k-1}$ in each identity test. The probability that the test works successfully for each edge is at least $1 - O(\log^2 n/n)$ by the union bound. We continue this for k rounds to find H_k, which is a perfect matching.

We need again $O(\log^2 n)$ random bits for the weight assignments $w_0, w_1, \ldots, w_{k-1}$ and the values for the x_i's. Note that we use the same random bits for x_i in all k rounds. This decreases the success probability, which is now at least $1 - O(\log^3 n)/n$ by the union bound.

In NC^2, we can construct the weight assignments and compute the determinants in each round. As we have $k = O(\log n)$ rounds, the overall complexity becomes NC^3.

Table of contents

Bipartite Planar Graphs

The SEARCH-PM problem already has some known NC-algorithms in the case of bipartite planar graphs [MN95, MV00, DKR10]. The one by Mahajan and Varadarajan [MV00] is in NC^3, while the other two are in NC^2. Our approach from the previous section can be modified to give an alternate NC^3-algorithm for this case.

The weights in our scheme in Section 3.2 become quasi-polynomial because we need to combine the different weight functions from $\log n$ rounds using a different scale. To solve this problem, we use the fact that in planar graphs, one can count the number of perfect matchings of a given weight in NC^2 by the Pfaffian orientation technique [Kas67, Vaz89]. As a consequence, we can actually construct the graphs G_i in each round in NC^2. Thereby we avoid having to combine the weight functions from different rounds.

In more detail, in the i-th round, we need to compute the union of minimum weight perfect matchings in G_{i-1} according to w_{i-1}. For each edge e, we decide in parallel if deleting e reduces the count of minimum weight perfect matchings. If yes, then edge e should be present in G_i. As it takes $\log n$ rounds to reach a single perfect matching, the algorithm is in NC^3.

Weighted perfect matchings and maximum matchings

A generalization of the perfect matching problem is the *weighted perfect matching problem* (WEIGHT-PM), where we are given a weighted graph, and we want to compute a perfect matching of minimum weight. There is no NC-reduction known from WEIGHT-PM to the perfect matching problem. However, the isolation technique works for this problem as well, when the weights are small integers. We put the given weights on a higher scale and put the weights constructed by our scheme in Section 3 on a lower scale. This ensures that a minimum weight perfect matching according to the combined weight function also has minimum weight according to the given weight assignment. Our scheme ensures that there is a unique minimum weight perfect matching. One can construct this perfect matching following the algorithm of Mulmuley et al. [MVV87] (Section 2.2).

Corollary 5.1. *For bipartite graphs,* WEIGHT-PM *with quasi-polynomially bounded integer weights is in* quasi-NC^2.

The maximum matching problem asks to find a maximum size matching in a given graph. It is well known that the maximum matching problem (MM) is NC-equivalent to the perfect matching problem (see for example [GKMT13]). The equivalence holds for both decision versions and the construction versions. The reductions also preserve bipartiteness of the graph. Thus, we get the following corollary.

Corollary 5.2. *For bipartite graphs,* MM *is in* quasi-NC^2.