Counting Shortest Two Disjoint Paths in Cubic Planar Graphs with an NC Algorithm

Andreas Björklund

Department of Computer Science, Lund University, Sweden

Thore Husfeldt

BARC, IT University of Copenhagen, Denmark and Lund University, Sweden

---
Abstract ──

Given an undirected graph and two disjoint vertex pairs s_1, t_1 and s_2, t_2, the Shortest two disjoint paths problem (S2DP) asks for the minimum total length of two vertex disjoint paths connecting s_1 with t_1, and s_2 with t_2, respectively.

We show that for cubic planar graphs there are NC algorithms, uniform circuits of polynomial size and polylogarithmic depth, that compute the S2DP and moreover also output the number of such minimum length path pairs.

Previously, to the best of our knowledge, no deterministic polynomial time algorithm was known for S2DP in cubic planar graphs with arbitrary placement of the terminals. In contrast, the randomized polynomial time algorithm by Björklund and Husfeldt, ICALP 2014, for general graphs is much slower, is serial in nature, and cannot count the solutions.

Our results are built on an approach by Hirai and Namba, Algorithmica 2017, for a generalisation of S2DP, and fast algorithms for counting perfect matchings in planar graphs.

1 Introduction

Shortest disjoint A, B-paths, introduced by Hirai and Namba [12], is the following problem: Let G be an undirected graph with two non-empty disjoint vertex subsets $A, B \subseteq V(G)$ of even size and an edge length function $\ell \colon E(G) \to \{1, \ldots, L\}$. An edge subset $E' \subseteq E(G)$ is a solution to *Disjoint A, B-paths* if it consists of $\frac{1}{2}(|A| + |B|)$ disjoint paths with endpoints both in A or both in B. The length $\ell(E')$ of a solution is $\sum_{e \in E'} \ell(e)$, and a *shortest* solution has length $\ell_{A,B} = \min_{E'} \ell(E')$. The objective is to compute $\ell_{A,B}$. The special case $|A| = |B| = 2$ is a well-studied problem called *Shortest two disjoint paths*.

We write $S_{A,B}$ for the number of solutions of length $\ell_{A,B}$. A graph is *cubic* (sometimes called 3-regular) if every vertex has degree 3. We prove the following:
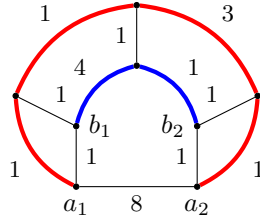
29th International Symposium on Algorithms and Computation (ISAAC 2018).
Editors: Wen-Lian Hsu, Der-Tsai Lee, and Chung-Shou Liao; Article No. 19; pp. 19:1–19:13

Leibniz International Proceedings in Informatics
LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Figure 1 A solution of minimum length $\ell_{A,B} = 11$ to Shortest disjoint A, B-paths with $A = \{a_1, a_2\}$ and $B = \{b_1, b_2\}$. Since $|A| = |B| = 2$, this is also an example of Shortest two disjoint paths. Note that neither path is a shortest path between its terminals.

▶ **Theorem 1.** *For any cubic planar n-vertex graph G, disjoint vertex subsets A and B, and edge length function $\ell : E(G) \to \{1, \dots, L\}$, we can compute $\ell_{A,B}$ and $S_{A,B}$ in deterministic $\tilde{O}(2^{|A\cup B|}n^{\omega/2+2}L^2)$ time[1], where $\omega < 2.373$ is the exponent of square matrix multiplication.*

In particular, for $|A| + |B| = O(1)$, the algorithm runs in deterministic time $\tilde{O}(n^{\omega/2+2}L^2)$.

To the best of our knowledge, no polynomial-time deterministic algorithm was known even for $|A| = |B| = 2$. Hirai and Namba's algorithm [12] works for general graphs in randomized time $n^{O(|A\cup B|)}$, so Theorem 1 also shows that cubic planar graphs allow better exponential dependency on $|A \cup B|$. In the appendix we show that all our algorithms extend to the case where the graph has maximum degree 3.

Because we can count the solutions we can use well-known techniques to retrieve a witness for the shortest length. By using our algorithm as a subroutine, we can retrieve the ith witness in a lexicographical order of the solutions by a polynomial overhead self reduction, by peeling off edges one at a time and remeasuring the number of solutions. In particular, by choosing i uniformly from $\{1, \dots, S_{A,B}\}$, we can sample uniformly over the solutions without first explicitly constructing the list of solutions.

Our algorithm is based on counting perfect matchings in a planar graph. Vazirani [30] showed how every bit in the number of perfect matchings in a planar graph can be decided by an NC algorithm, i.e., uniform polylogarithmically shallow polynomial size circuits, an observation he attributes to Luby. Using his algorithm as a subroutine, we arrive at an efficient parallel algorithm. We state the result for Shortest *two* disjoint paths:

▶ **Theorem 2.** *For any cubic planar n-vertex graph G, disjoint vertex subsets A and B with $|A| = |B| = 2$, and edge length function $\ell : E(G) \to \{1, \dots, L\}$, we can compute $\ell_{A,B}$ and $S_{A,B}$ by an NC algorithm.*

The same statement holds as long as $|A| + |B|$ is logarithmic in n.

Via the Isolation lemma of Mulmuley *et al.* [23] we can also obtain a witness, i.e., a solution E' of length $\ell(E') = \ell_{A,B}$, with a *randomized* NC algorithm. We note that the recent breakthrough result showing how to find a perfect matching in a planar graph in NC by Anari and Vazirani [2] doesn't seem to be directly applicable to our problem. Our algorithm counts the solutions to Shortest two disjoint paths by an annihilation sieve, i.e. the number of solutions is an alternating sum of perfect matchings in a set of graphs, but many of the terms cancel each other. Hence there are many perfect matchings that do not correspond to a solution. Finding one deterministically won't help us.

We also provide evidence that the exponential dependence on $|A| + |B|$ is necessary:

---

[1] The $\tilde{O}(f(n))$ notation suppresses factors polylogarithmic in $f(n)$.

▶ **Theorem 3.** *It is #P-hard to simultaneously compute the length and the number of solutions to Shortest disjoint A, B-paths in cubic planar graphs.*

1.1 Hirai and Namba's Result

Hirai and Namba [12] shows that Shortest disjoint A, B-paths has a randomized algorithm running in $n^{O(|A \cup B|)}$ time, that w.h.p. finds the length of the shortest disjoint paths. Their algorithm is inspired by the algorithm of Gallai [9] that can be used to address the special case $B = \emptyset$, and the algorithm by Björklund and Husfeldt [4] for the special case $|A| = |B| = 2$. They apply a two-step method: First, expand G into another edge weighted graph G' using so-called Gallai paths, in a way that the weighted perfect matchings in G' can be used to obtain the solution to the original problem. Then it uses the fact that counting perfect matchings in G' modulo 2^k has a $n^{O(k)}$ time algorithm. In their reduction, each solution is counted $2^{|A \cup B|/2}$ times, so they need to set $k > \frac{1}{2}|A \cup B|$ to count something meaningful, but still small enough to keep the running time down. Thus, their algorithm is capable only of counting the solutions modulo a fixed small power of 2. The Isolation lemma [23] ensures that the number of solutions is not divisible by this small power of two. This is why [12] need randomness; the same problem appears in the shortest paths algorithm of [4].

1.2 Our Approach

We apply Hirai and Namba's approach to the planar cubic case. It is well-known that in any planar graph we can count the perfect matchings in polynomial time. In particular we don't just obtain the result modulo a small power of two. We use this, but there is one obstacle that needs to be addressed to accomplish this: The reduction Hirai and Namba use does not preserve planarity. Our contribution is to show that for cubic planar graphs, we can construct a set of $2^{|A \cup B|/2}$ cubic planar graphs, each having non-negative edge weights, so that a linear combination of the number of weighted perfect matchings in these graphs can be used to deduce the number of solutions to the Shortest disjoint A, B-paths in the original instance. We are inspired by the result of Galluccio and Loebl [10] that shows how to count perfect matchings in graphs of genus g by constructing 4^g orientations and computing the Pfaffian for each of them. We choose to use a more direct approach instead of reducing to their result to make the description of our algorithm more self contained.

1.3 Related Work

Björklund and Husfeldt [4] showed that Shortest two disjoint paths in a general unweighted undirected graph has a polynomial time Monte Carlo algorithm. Colin de Verdiére and Schrijver [8], and Kobayachi and Sommer [19] showed that for planar graphs, deterministic polynomial-time algorithms for the Shortest two disjoint paths exist if the four terminals lie on the boundary of at most two faces. The algorithm in the present paper works no matter where the terminals are, but is much slower. Still, it is significantly faster than the general $O(n^{11})$ time algorithm by Björklund and Husfeldt [4].

Very recently, Datta et al. [7] presented a deterministic algorithm independently of ours for Shortest k-disjoint paths in planar graphs conditioned on the terminals either all being placed on the same face or all source terminals on one face and the target terminals on another. In particular, restricted to the Shortest two disjoint path problem, their algorithm does not work for arbitrary placement of the terminals as ours does. Interestingly, their algorithm is also based on computing determinants just as ours and can count the solutions just as our algorithms can, although they don't use Pfaffian orientations as we do.

For the decision problem of detecting two disjoint paths joining given vertex pairs, no matter their length, deterministic polynomial-time algorithms have been known since 1980 for general graphs, by Ohtsuki [24], Seymour [25], Shiloah [26], and Thomassen [28]; all published independently. Tholey [27] reduced the running time for that problem to near-linear. Khuller *et al.* [17] showed that the problem can be solved in NC.

The k-disjoint paths problem is the natural generalisation of the two disjoint paths problem: Given a list $\{(s_1, t_1), \ldots, (s_k, t_k)\}$ of terminal pairs, decide if there exist k disjoint paths connecting s_i with t_i for $i \in \{1, \ldots, k\}$. Again neglecting the length of the solution, this problem has a polynomial time algorithm in general graphs for fixed k, but the dependence on k is horrible ($\exp \exp \exp \exp O(k)$, see [16]). For planar graphs, there exists a doubly exponential ($\exp \exp O(k)$) $\text{poly}(n)$ time algorithm, by Adler *et al.* [1]. For comparison, our running time dependence is singly exponential in the size of the terminal set, but of course our criteria for allowed connections is much relaxed.

The special case $B = \emptyset$ in Disjoint A, B-paths is referred to as A-Paths in Lovász and Plummer [22]. Its solution in general undirected graphs by a polynomial time algorithm was given by Gallai [9] by a reduction to finding a perfect matching. Using Mulmuley, Vazirani, and Vazirani's algorithm for the problem they call Exact Matching [23] on Gallai's construction, one can in randomized polynomial time solve the Shortest disjoint A-paths.

The idea of using perfect matching counting in restricted graph classes to solve other combinatorial optimisation problems is not new, a classic example is the polynomial time algorithm for Max Cut in graphs of bounded genus by Galluccio, Loebl, and Vondrák [11].

2   Algorithmic Results: Theorems 1 and 2

2.1   Notation

We consider an undirected graph G with vertex set $V = V(G)$ and edge set $E = E(G)$. A (u, v)-path is a path from vertex u to vertex v. A *perfect matching* in G is a subset $E' \subseteq E$ of the edges of size $|E'| = \frac{1}{2}|V|$, such that every vertex in $v \in V$ is the endpoint of exactly one edge in E'. Let $w \colon E \to \mathbf{N}$ be an edge weight function to positive integers. Let $\mathcal{M}(G)$ be the family of perfect matchings in G. We denote by $\text{pm}(G)$ the sum of the weighted perfect matchings in a graph G, i.e.,

$$\text{pm}(G) = \sum_{M \in \mathcal{M}(G)} \prod_{e \in M} w(e).$$

If the weights are unity, this is the number of perfect matchings. In our algorithm's analysis, some edges are weighted by an indeterminate s and $\text{pm}(G)$ is a polynomial in s. However, the algorithm itself only works directly over the integers after replacing the indeterminate s for a numerical value.

2.2   Pfaffian Orientations

A *Pfaffian orientation* of a graph G with edge weights $w \colon E \to \mathbf{N}$, is an orientation of the edges $q \colon E \to \{-1, 1\}$ so that the skew-symmetric adjacency matrix A_G, where

$$\forall uv \in E, u < v \colon q(u, v)w(uv) = A_G(u, v) = -A_G(v, u) = -q(u, v)w(uv),$$
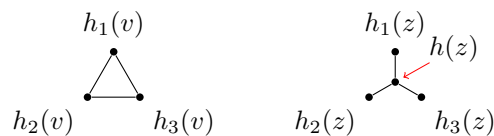
satisfies

$$\text{pm}(G) = \left| \sqrt{\det(A_G)} \right|. \tag{1}$$

An orientation of a graph G is Pfaffian if and only if every even-length cycle C such that $G \setminus V(C)$ has a perfect matching, has an odd number of edges directed in either direction along C. Kasteleyn [15], famously proved that all planar graphs have a Pfaffian orientation, and moreover showed how you given a planar graph can find a Pfaffian orientation fast. Nowadays it is even known how to find one in planar graphs in linear time, and Vazirani [30] showed it can be computed in NC. In general it only holds that $|\mathrm{pm}(G)| = \left| \sqrt{\det(A_G)} \right|$, but we only consider positive edge weights in this paper and hence already know $\mathrm{pm}(G)$ to be non-negative. Little [21] extended Kasteleyn's method to also work constructively for graphs that do not have a $K_{3,3}$ subgraph as a minor. However, cubic $K_{3,3}$ minor free graphs coincide with the set of cubic planar graphs.

2.3 Reduction from Disjoint A,B-Paths to Counting Perfect Matchings

Consider as input a cubic planar graph G and let $\ell \colon E \to \{1, \ldots, L\}$ be an edge length function, along with two disjoint subsets A and B of the vertices, each having even size. Set $\Lambda = \sum_{e \in E} \ell(e)$. We will reduce Shortest disjoint A, B-paths to counting perfect matchings so that planarity is preserved. In this section, we write Z for the set of terminals, $Z = A \cup B$.

We build a larger graph H from G as follows. Replace each nonterminal $v \in V \setminus Z$ with three vertices $h_1(v)$, $h_2(v)$, and $h_3(v)$ forming a triangle. Replace each terminal $z \in Z$ by a 3-star on vertices $h_1(z)$, $h_2(z)$, $h_3(z)$, and *terminal center* $h(z)$. The gadgets look like this:

We call the edges within these two gadgets *internal* edges.

Moreover, if $uv \in E(G)$, then $h_i(u)h_j(v)$ is also an edge in H for some $i, j \in \{1, 2, 3\}$ in such a way that each vertex in H is used in exactly one of the additional edges. We call these edges in H between gadgets *external* edges. We write $f(uv) = h_i(u)$ and $g(uv) = h_j(v)$ to identify the two gadget vertices in H connected by the external edge representing uv. Confer figure 2. The graph H has the property that every vertex except the terminal centers $h(z)$ for $z \in Z$ is part of exactly one external edge. Our first insight is the following:
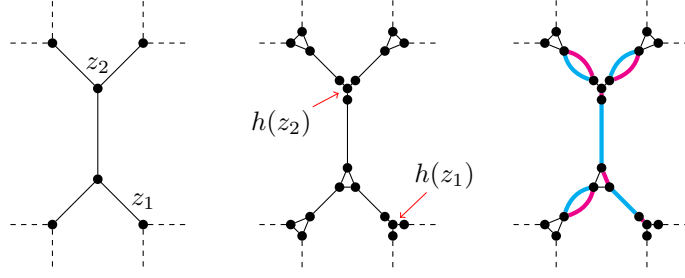
▶ **Lemma 4.** *If G is planar, then so is H.*

Proof. Both gadgets are easily seen to be planar. To see that H is planar, use an embedding of G. For each vertex v in G, consider a small enough circle C_v around v containing no other edge or vertex. Now replace v with a copy of its gadget small enough to fit C_v. ◀

Hence, if G is planar, we can find a Pfaffian orientation of H, as well as for any subgraph of it, as any subgraph is also planar. (We note in passing that a Pfaffian orientation of a graph is not necessarily a Pfaffian orientation of its subgraph.)

From H, we create several graphs depending on a subset of the terminal vertices. We write $H(X)$ for $X \subseteq Z$ to mean the graph obtained from H by removing the terminal centers $h(z)$ and all incident edges for each terminal $z \notin X$. We have $H = H(Z)$.

We introduce an indeterminate s to control the length of the paths. We write $D(X, s)$ for a skew-symmetric adjacency matrix of a Pfaffian orientation of $H(X)$, where we have multiplied all entries representing an external edge e in $H(X)$ with $s^{\ell(e)}$.

Our algorithm is a direct application of the following result:

Figure 2 Left: The instance graph G with two terminal nodes z_1 and z_2. Middle: The gadget graph H. Right: The union of two matchings in $H(X)$ and $H(Z \setminus X)$ for some X containing both z_1 and z_2. Together, they form a path between $h(z_1)$ and $h(z_2)$, along with three double edges.

▶ **Lemma 5.** *For a graph G, consider*

$$p(G, s) = \sum_{X \subseteq Z} (-1)^{|X \cap A|} \left| \sqrt{\det(D(X, s)) \det(D(Z \setminus X, s))} \right| \tag{2}$$

as a polynomial in the indeterminate s. Let cs^d be the largest degree monomial with a positive coefficient in $p(G, s)$. Then, $\ell_{A,B} = 2\Lambda - d$ is the shortest total length of any disjoint A, B-paths in G, and c is $2^{|Z|/2}$ times the number of solutions having that minimum length.

Proof. We begin by arguing that $p(G, s)$ indeed is a polynomial in the indeterminate s. Fix $X \subseteq Z$. Write $\mathcal{M}(X)$ for the set $\mathcal{M}(H(X))$ of perfect matchings in $H(X)$. By (1), we have

$$\left| \sqrt{\det(D(X, s))} \right| = \mathrm{pm}(H(X)) = \sum_{M \in \mathcal{M}(X)} \prod_{e \in M} w(e) \,,$$

where $w(e) = s^{\ell(e)}$ if e is external and $w(e) = 1$ if e is internal. Thus, for a pair of perfect matchings $M_1 \in \mathcal{M}(X)$ and $M_2 \in \mathcal{M}(Z \setminus X)$, their contributing term $t(M_1, M_2)$ is

$$t(M_1, M_2) = \left(\prod_{e \in M_1} w(e) \right) \cdot \prod_{e \in M_2} w(e) \,,$$

which is clearly a polynomial in s, and write

$$p(G, s) = \sum_{X \subseteq Z} (-1)^{X \cap A} \sum_{M_1 \in \mathcal{M}(X)} \sum_{M_2 \in \mathcal{M}(Z \setminus X)} t(M_1, M_2) \,.$$

Now view $M_1 \cup M_2$ as a subgraph in H, by identifying each vertex in $H(X)$ and $H(Z \setminus X)$ with its copy in H. (It is helpful to view $M_1 \cup M_2$ as a multiset, so the corresponding subgraph is in fact a multigraph using the edges $M_1 \cap M_2$ twice.) We can visualise this as placing the two graphs on top of each other and looking at the subgraph formed by the two matchings. It is clear that every vertex in H has degree at most 2 in this subgraph, so $M_1 \cup M_2$ can be partitioned into three edge subsets $\mathcal{P}, \mathcal{C}, \mathcal{D} \subseteq E(H)$, such that $\mathcal{P}$ is a disjoint union of simple paths, $\mathcal{C}$ is a disjoint union of simple cycles, and $\mathcal{D}$, which is equal to the intersection $M_1 \cap M_2$, is a disjoint union of isolated edges.

We claim that every path in $\mathcal{P}$ has its endpoints in terminal centers. To see this, first note that each terminal centre $h(z)$ for $z \in Z$ is present in exactly one of the graphs $H(X)$ and $H(Z \setminus X)$. Therefore, $h(z)$ is matched by exactly one edge in $M_1 \cup M_2$ and therefore is the endpoint of a path. Every other vertex in H appears in both $H(X)$ and $H(Z \setminus X)$ and is therefore matched in both M_1 and M_2; in particular, no such vertex is the endpoint of a simple path. Figure 2 shows a small example.

We next argue that unions $M_1 \cup M_2$ whose paths connect terminal centers $h(a)$ and $h(b)$ with $a \in A$ and $b \in B$ contribute nothing to $p(G, s)$. To this end, consider such a term $t(M_1, M_2)$ with $M_1 \in \mathcal{M}(X)$ and $M_2 \in \mathcal{M}(Z \setminus X)$ and let $P = (u_1, \ldots, u_k)$ with $u_1 = h(a), u_k = h(b)$ be the lexicographically first such path in $\mathcal{P}$.

If k is odd, then the edges u_1u_2, u_3u_4, ..., $u_{k-2}u_{k-1}$ belong to one matching, say M_1, and the edges u_2u_3, ..., $u_{k-1}u_k$ belong to M_2. In particular, the terminal center $h(a)$ is matched in M_1, which implies $h(a) \in V(H(X))$ and therefore $a \in X$. Conversely, $h(b)$ is matched in M_2, which implies $h(b) \in V(H(Z \setminus X))$ and $b \notin X$. Now form $X' = (X \cup \{b\}) \setminus \{a\}$ and consider the two matchings $M_1' \in \mathcal{M}(X')$ and $M_2' \in \mathcal{M}(Z \setminus X')$ created from M_1 and M_2 by swapping the edges on P. Note that the edge $u_{k-1}u_k$ incident on $h(b)$ now belongs to M_1', and since b belongs to X', the matching M_1' is indeed a perfect matching in $\mathcal{M}(X')$. Similarly, $M_2' \in \mathcal{M}(Z \setminus X')$. Starting the exact same process from the matchings M_1' and M_2' and set X' would get us back to M_1, M_2, and X, since the same path P will be chosen by the lexicographical order, so the process defines a fixed-point free involution on the set of terms $t(M_1, M_2)$ and subsets of Z. Crucially, the contribution to (2) of terms paired by this involution cancel:

$$(-1)^{|X \cap A|}t(M_1, M_2) + (-1)^{|X' \cap A|}t(M_1', M_2') = 0\,,$$

because the multisets $M_1 \cup M_2$ and $M_1' \cup M_2'$ are the same, and X' and X differ in exactly one terminal from A. Hence no such terms survive in the computation of $p(G, s)$.

If k is even, then u_1u_2, u_3u_4, ..., $u_{k-1}u_k$ belong to the same matching, say M_1. Thus, both $h(a)$ and $h(b)$ belong to $H(X)$, so a and b belong to X. Set $X' = X \setminus \{a, b\}$, and follow the same argument as above.

In other words, $t(M_1, M_2)$ survives in $p(G, s)$ only if the disjoint paths in $\mathcal{P}$ have their endpoints either both in A or both in B. The contribution is

$$t(M_1, M_2) = \left(\prod_{e \in \mathcal{D}} w(e)^2 \right) \cdot \prod_{e \in \mathcal{C} \cup \mathcal{P}} w(e) = s^d\,,$$

where

$$d = \left(2 \sum_{e \in \mathcal{D}} \ell(e) \right) + \sum_{e \in \mathcal{C} \cup \mathcal{P}} \ell(e) = 2\Lambda - \sum_{e \in \mathcal{C} \cup \mathcal{P}} \ell(e)\,,$$

with the convention that $\ell(e) = 0$ for internal edges. The last term is at least $\ell_{A,B}$, and attains that value exactly if $\mathcal{C}$ is empty and $\mathcal{P}$ contains the external edges of a solution E' to Shortest disjoint A, B-paths in G. Otherwise, $d < 2\Lambda - \ell_{A,B}$.

We finally turn to the other direction, to show that if there exists disjoint A, B-paths in G, we will detect them in $p(G, s)$. Moreover, we argue that we can count the ones of shortest total length. To see this, first consider a solution $E' \subseteq E(G)$ to Shortest disjoint A, B-paths, i.e., a disjoint union of paths

$$E' = P_1 \cup \cdots \cup P_{|Z|/2}\,,$$

each of which has terminal endpoints either both in A, or both in B. Let T be a subgraph of H obtained in the following way. For each such path $P = (v_1, \ldots, v_k)$, first add the external edges $f(v_1v_2)g(v_1v_2)$, ..., $f(v_{k-1}v_k)g(v_{k-1}v_k)$ to T. Second, add the internal edges $h(v_1)f(v_1v_2)$ and $g(v_{k-1}v_k)h(v_k)$ in the two terminal gadgets, and the internal edges $g(v_iv_{i+1})f(v_{i+1}v_{i+2})$ in the nonterminal gadgets for $i \in \{1, \ldots, k-2\}$. This adds precisely one internal edge per gadget representing a vertex on P. Third, for every vertex $u \in V(H)$

not used in an edge so far, we add to T its unique external edge in H. This is where we use the property of H that every non-terminal vertex has a unique external edge. Thus, T consists of disjoint edge sets $\mathcal{P}, \mathcal{D} \subseteq E(H)$ where $\mathcal{P}$ consists of disjoint paths and $\mathcal{D}$ consists of disjoint (external) edges.

We continue to account for the contribution of T to (2). Let $X \subseteq Z$ be a subset of terminals such that the endpoints of the paths in $\mathcal{P}$ are either both in X or both in $Z \setminus X$. In particular, $|X \cap A|$ is even, and there are $2^{|Z|/2}$ such subsets. There is exactly one perfect matching M_1 in $H(X)$ that is a subgraph of T; this matching contains all the internal edges on the paths of $\mathcal{P}$ with endpoints both in X. There is also exactly one perfect matching M_2 in $H(Z \setminus X)$ that is a subgraph of T; this matching contains all the external edges on the paths of $\mathcal{P}$ with endpoints both in $Z \setminus X$. In particular, every external edge in $\mathcal{D}$ appears exactly twice in the multiset $M_1 \cup M_2$, and every external edges in $\mathcal{P}$ appears exactly once. (The internal edges have weight 1, so we need not count their contribution to a product.) Thus, the total contribution of M_1 and M_2 is

$$ t(M_1, M_2) = \left(\prod_{e \in \mathcal{D}} w(e)^2 \right) \cdot \prod_{e \in \mathcal{P}} w(e) = s^d, \quad \text{where } d = 2\Lambda - \ell_{A,B}, $$

and the solution E' accounts for the contribution

$$ \sum_{X \subseteq Z} (-1)^{|X \cap A|} t(M_1, M_2) = 2^{|Z|/2} s^d. $$

All other surviving terms have lower degree in $p(G, s)$, and the lemma follows. ◀

2.4 Algorithm

Our algorithm computes the coefficients of $p(G, s)$ in (2) viewed as a polynomial in s, using polynomial interpolation. The algorithm works through direct evaluation in sufficiently many points $s \in \{0, 1, \ldots, 2\Lambda\}$ of $p(G, s)$ after replacing s for its numerical value. Hence all computations are over the integers.

  1. For $s = 0$ to 2Λ,
  2.   Set $sum_s = 0$.
  3.   For $X \subseteq Z$, $|X|$ even,
  4.   Construct $H(X)$ and $H(Z \setminus X)$ and their Pfaffian orientations.
  5.   Compute the integers $\det^2(D(X))$ and $\det^2(D(Z \setminus X))$ for the current value of s.
  6.   Take the fourth root of the two determinants and multiply them.
  7.   Add the product with the sign $(-1)^{|X \cap A|}$ to sum_s.
  8. Use polynomial interpolation to compute the coefficients of $p(G, s)$ from the array sum.
  9. Locate the largest non-zero monomial cs^d.
10. Return $\ell_{A,B} = 2\Lambda - d$ and $S_{A,B} = c/2^{|Z|/2}$.

2.5 Sequential Running Time

We turn to the sequential running time of the algorithm from section 2.4. Recall that the polynomial $p(G, s)$ has degree at most 2Λ, and hence the number of evaluated points is sufficient to uniquely recover its coefficients. We can bound the value of the two determinants using the Leibniz formula for the determinant. There are at most $3^{3n+|A \cup B|}$ terms since there are at most 3 choices per vertex in $H(X)$. For each choice, the largest value is obtained when the external edges are picked twice, $i.e.$, every term is at most $(2\Lambda)^{2\Lambda}$. Hence the

determinant can be a $\beta = \tilde{O}(nL)$ bit number. We can compute the determinants in row 5 using $O(n^{\omega/2})$ arithmetic operations, using Yuster's algorithm [32] for the square of the determinant, which in turn uses the dissection method developed by Lipton *et al.* [20]. Note that since we know all our determinants to be positive (each is the square of the number of perfect matchings), no information is lost by computing even powers of the determinant. Every arithmetic operation can be computed in $\tilde{O}(\beta)$ time [31]. Computing all determinants requires at most $\tilde{O}(\Lambda 2^{|A \cup B|} n^{\omega/2}\beta) = \tilde{O}(2^{|A \cup B|} n^{\omega/2+2} L^2)$ time. This part dominates the computation time, since taking the square roots in row 7 using Newton's method requires only about $\log nL$ iterations for an integer square, and the polynomial interpolation in row 8 can be done in quadratic time. It requires $\tilde{\Omega}(\Lambda)$ operations over a finite field, cf. [31], and we need a field, or several fields and the Chinese remainder theorem, of total size $\Omega(\beta)$ to recover the integer values. This completes the proof of Theorem 1.
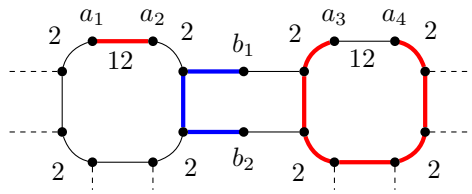
2.6 Parallel Circuit Depth

In this section we prove that our algorithm in section 2.4 can be efficiently implemented as a circuit of polynomial size and polylogarithmic depth. First we note that all values of s and all values of X in row 1 and 3 of the algorithm can be evaluated in parallel. All computations are made on integers of $\beta = \tilde{O}(nL)$ bits as claimed in the previous section. Addition and multiplication on β bit integers can be done in $\text{polylog}(\beta)$ depth. Constructing the graphs $H(X)$ in row 4 can be done even without a planar embedding of G, as it doesn't matter how the external edges are mapped to the gadget's connectors, planarity is always preserved. Vazirani shows that the number of perfect matchings can be computed by an NC algorithm [30], see also the textbook [14]. He describes how a Pfaffian orientation for a planar graph can be obtained via Klein and Reif's parallel planar embedding algorithm [18]. He next uses the fact that the determinant can be computed in NC, a consequence of Csansky's algorithm for the determinant [6]. Berkowitz algorithm [3] via iterated matrix product can also be used (see Cook [5]). Computing the integer square root at row 6 is a logarithmic depth task with Newton's method since the convergence is quadratic. Once all evaluations are done, the inner loop summation at row 7 can be computed for all s, again in polylogarithmic depth by a balanced binary tree of adders of β-sized integers. Finally, Cook describes how polynomial interpolation is in NC [5] by reducing to Berkowitz algorithm for the determinant [3]. This completes the proof of Theorem 2.

3 Hardness Result: Theorem 3

Our hardness reduction is from counting maximum independent sets in cubic planar graphs, proven #P-hard in Vadhan [29] (Corollary 4.2.1). The NP-hardness result for Disjoint A, B-paths in general graphs by Hirai and Namba [12], follows Hirai and Pap [13]. It is a reduction directly from 3-Satisfiability but it is not (weakly) parsimonious. We give here such a strengthened reduction.

Consider a cubic planar graph G in which we want to count the maximum independent sets. From G, we construct a maximum degree 3 planar instance I to Shortest disjoint A, B-paths. As described in the previous section, we can by adding a few vertices per vertex of degree less than three make sure the graph is cubic while preserving planarity. Here we will stick with a few vertices of degree two in our description of I for simplicity. First, for every vertex $v \in V$, we add a clockwise ordered cycle $v'_1, \ldots, v'_8$. The edge $v'_8 v'_1$ has length 12 whereas all other edges $v'_i v'_{i+1}$ for $i \in \{1, \ldots, 7\}$ have length 2 if i is odd and length 1 if i is even. Furthermore, vertices v'_1 and v'_8 belong to A for every vertex v. Second, for every edge $uv \in E$, we add two vertices w'_1 and w'_2 to I. We add edges $w'_1 u'_i, w'_2 u'_{i+1}, w'_1 v'_j$, and

Figure 3 Two vertex gadgets and one edge gadget in the constructed instance I in the #P-hardness proof. The edge terminals in B must connect through some vertex gadget, forcing the A terminals on it to connect through the longer length 12 edge alternative.

$w_2'v_{j-1}$ of length 1 for some indices i and j so that no vertex is used more than once, and the resulting graph I is planar. This is easy to accomplish by using a planar embedding of G and order edges incident on a vertex in clockwise order. See Figure 3. Furthermore, we add w_1' and w_2' to B.

▶ **Lemma 6.** *Let $\ell_{A,B}$ and $S_{A,B}$ be the solution to Shortest disjoint A, B-paths on I, then the maximum independent set in G has size $\alpha(G) = 12|V| + 3|E(G)| - \ell_{A,B}$ and the number of such sets are $S_{A,B}/2^{|E(G)|-3\alpha(G)}$.*

Proof. Any vertex pair in A on the same vertex gadget cycle must be connected with each other through a path, since there are no paths between different vertex gadget cycles that do not also pass through a terminal in B. Hence there are only two possibilities for every such pair: either it is connected through the 12-long edge between them, or it uses the path around the cycle of length 11. Let $I \subseteq V$ be the set of vertices whose vertex gadgets uses paths of length 11 to connect its two A terminals. The set I must be an independent set in G, since the terminals on every edge gadget must use some edge on either of the two vertex gadgets it is connected to. Moreover, any pair of terminals in B cannot be connected with a path shorter than 3 as there exist no such short paths between any pair of them. A lower bound on the attainable length of a Shortest disjoint A, B-paths solution is hence $12|V| - \alpha(G) + 3|E|$, where $\alpha(G)$ is the size of a maximum independent set in G. Any such solution can naturally be interpreted as a maximum independent set in G by identifying the A-paths of length 11.

Moreover, from any maximum independent set I in G, we can construct disjoint paths of this length, simply by taking the 12-long edge for every vertex not in I for a vertex gadget's A terminals, and the shorter 11-long path for the other vertex gadgets. The edge gadgets' B terminals can be connected pairwise with each other through a 3-long path using an edge on either of its two adjacent vertex gadgets, whenever it represents a vertex not in I. It might be possible to connect the B terminals in other ways, but those paths will be of length strictly longer than 3 as they need to use a 2-long edge on some vertex gadget. There are precisely $|E| - 3\alpha(G)$ edges with neither endpoint in I, and hence the maximum independent sets will be counted $2^{|E|-3\alpha(G)}$ times in the Shortest disjoint A, B-paths.   ◀

Theorem 3 now directly follows from Lemma 6, since if we can find $\ell_{A,B}$ in I, we can also compute the number of maximum independent sets in G from $S_{A,B}$.

── **References** ──────────────────────────────────

1   Isolde Adler, Stavros G. Kolliopoulos, and Dimitrios M. Thilikos. Planar Disjoint-Paths Completion. In *Parameterized and Exact Computation - 6th International Symposium, IPEC 2011, Saarbrücken, Germany, September 6-8, 2011.*, pages 80–93, 2011. `doi:10.1007/978-3-642-28050-4_7`.

2 Nima Anari and Vijay V. Vazirani. Planar Graph Perfect Matching is in NC. *CoRR*, abs/1709.07822, 2017. `arXiv:1709.07822`.

3 Stuart J. Berkowitz. On Computing the Determinant in Small Parallel Time Using a Small Number of Processors. *Inf. Process. Lett.*, 18(3):147–150, 1984. `doi:10.1016/0020-0190(84)90018-8`.

4 Andreas Björklund and Thore Husfeldt. Shortest Two Disjoint Paths in Polynomial Time. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 211–222, 2014. `doi:10.1007/978-3-662-43948-7_18`.

5 Stephen A. Cook. A Taxonomy of Problems with Fast Parallel Algorithms. *Information and Control*, 64(1-3):2–21, 1985. `doi:10.1016/S0019-9958(85)80041-3`.

6 L. Csanky. Fast Parallel Matrix Inversion Algorithms. *SIAM J. Comput.*, 5(4):618–623, 1976. `doi:10.1137/0205040`.

7 Samir Datta, Siddharth Iyer, Raghav Kulkarni, and Anish Mukherjee. Shortest k-Disjoint Paths via Determinants. *CoRR*, abs/1802.01338, 2018. Accepted to FSTTCS 2018. `arXiv:1802.01338`.

8 Éric Colin de Verdière and Alexander Schrijver. Shortest vertex-disjoint two-face paths in planar graphs. *ACM Trans. Algorithms*, 7(2):19:1–19:12, 2011. `doi:10.1145/1921659.1921665`.

9 Tibor Gallai. Maximum-minimum Sätze und verallgemeinerte Faktoren von Graphen. *Acta Mathematica Academiae Scientiarum Hungaricae*, 12:131–173, 1961.

10 Anna Galluccio and Martin Loebl. On the Theory of Pfaffian Orientations. I. Perfect Matchings and Permanents. *Electr. J. Comb.*, 6, 1999. URL: `http://www.combinatorics.org/Volume_6/Abstracts/v6i1r6.html`.

11 Anna Galluccio, Martin Loebl, and Jan Vondrák. Optimization via enumeration: a new algorithm for the Max Cut Problem. *Math. Program.*, 90(2):273–290, 2001. `doi:10.1007/PL00011425`.

12 Hiroshi Hirai and Hiroyuki Namba. Shortest (A+B)-Path Packing Via Hafnian. *Algorithmica*, 80(8):2478–2491, 2018. `doi:10.1007/s00453-017-0334-0`.

13 Hiroshi Hirai and Gyula Pap. Tree metrics and edge-disjoint S-paths. *Math. Program.*, 147(1-2):81–123, 2014. `doi:10.1007/s10107-013-0713-5`.

14 Marek Karpinski and Wojciech Rytter. *Fast parallel algorithms for graph matching problems*. Oxford University Press Inc. New York, NY, USA, 1998.

15 Pieter W. Kasteleyn. Graph Theory and Chrystal Physics. In F. Harary, editor, *Graph Theory and Chrystal Physics*, pages 47–52. Academic Press, London, 1957.

16 Ken-ichi Kawarabayashi and Paul Wollan. A shorter proof of the graph minor algorithm: the unique linkage theorem. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Mass., USA, 5-8 June 2010*, pages 687–694, 2010. `doi:10.1145/1806689.1806783`.

17 Samir Khuller, Stephen G. Mitchell, and Vijay V. Vazirani. Processor Efficient Parallel Algorithms for the Two Disjoint Paths Problem and for Finding a Kuratowski Homeomorph. *SIAM J. Comput.*, 21(3):486–506, 1992. `doi:10.1137/0221032`.

18 Philip N. Klein and John H. Reif. An Efficient Parallel Algorithm for Planarity. *J. Comput. Syst. Sci.*, 37(2):190–246, 1988. `doi:10.1016/0022-0000(88)90006-2`.

19 Yusuke Kobayashi and Christian Sommer. On shortest disjoint paths in planar graphs. *Discrete Optimization*, 7(4):234–245, 2010. `doi:10.1016/j.disopt.2010.05.002`.

20 Richard J. Lipton, Donald J. Rose, and R. E. Tarjan. Generalized nested dissection. *SIAM Journal on Numerical Analysis*, 16(2):346–358, 1979.
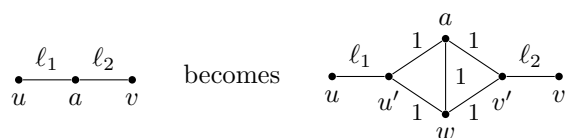
21   Charles H. C. Little. An extension of Kasteleyn's method of enumerating the 1-factors of planar graphs. In D. Holton, editor, *Combinatorial Mathematics, Proceedings 2nd Australian Conference*, number 403 in Lecture Notes in Mathematics, pages 63–72, 1974.

22   László Lovász and Michael D. Plummer. *Matching Theory*. North Holland, 1986.

23   Ketan Mulmuley, Umesh V. Vazirani, and Vijay V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7(1):105–113, 1987.

24   T. Ohtsuki. The two disjoint path problem and wire routing design. In *Graph Theory and Algorithms, Proc. 17th Symposium of Research Institute of Electric Communication (Sendai, Japan, October 24–25, 1980)*, pages 207–216. Springer, 1981.

25   Paul D. Seymour. Disjoint paths in graphs. *Discrete Mathematics*, 29(3):293–309, 1980. `doi:10.1016/0012-365X(80)90158-2`.

26   Yossi Shiloach. A Polynomial Solution to the Undirected Two Paths Problem. *J. ACM*, 27(3):445–456, 1980. `doi:10.1145/322203.322207`.

27   Torsten Tholey. Solving the 2-Disjoint Paths Problem in Nearly Linear Time. *Theory Comput. Syst.*, 39(1):51–78, 2006. `doi:10.1007/s00224-005-1256-9`.

28   Carsten Thomassen. 2-linked graphs. *Eur. J. Combin*, 1:371–378, 1980.

29   Salil P. Vadhan. The Complexity of Counting in Sparse, Regular, and Planar Graphs. *SIAM J. Comput.*, 31(2):398–427, 2001.

30   Vijay V. Vazirani. NC algorithms for computing the number of perfect matchings in $K_{3,3}$-free graphs and related problems. *Information and Computation*, 80:152–164, 1989.

31   Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra (3. ed.)*. Cambridge University Press, 2013.

32   Raphael Yuster. Matrix Sparsification for Rank and Determinant Computations via Nested Dissection. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 137–145, 2008. `doi:10.1109/FOCS.2008.14`.

A   Appendix

We extend the algorithm to planar graphs of *maximum* degree 3. First consider an edge ua where u is a nonterminal vertex of degree 3 and a is a terminal of degree 1. Then ua can be removed and u inserted into the terminal set of a; the resulting instance has a shortest solution of size $\ell_{A,B} - \ell(ua)$. When u is also a terminal vertex, there are two cases: If u belongs to the same terminal set as a then ua must be a path in the shortest solution, so we can remove both u and a and discount the resulting value by $\ell(ua)$. If u belongs to the other terminal set than a then there is no solution and we can output $S_{A,B} = 0$.

Consider a u, v-path P whose internal vertices all have degree 2. If none of P's internal vertices are terminals then P can be contracted into a single edge with the sum of the original edge lengths. If P contains alternating terminals, say $a \in A$, $b \in B$, $a' \in A$ in that order, no solution can exist. If P contains exactly two terminals $a \in A$ and $b \in B$ then its prefix from u to a can be contracted into a single edge, and so can its suffix from b to v; the infix from a to b can be removed. The resulting dangling edges ua and vb are handled as above.

In general, we can replace a degree-2 terminal a incident on the edges ua and av with the 4-vertex 'diamond' graph, introducing 3 new nonterminal vertices. The original edges retain their lengths, and the new edges receive length 1, so that

No path with endpoint a in an optimal solution uses w, because $uu'a$ is shorter than $uu'wa$. No other solution uses the nonterminal w either, because this would isolate a. Thus, an optimal solution uses either $au'u$ or $av'v$ and no other edges in the gadget. We conclude that every optimal solution in the transformed graph corresponds to exactly one optimal solution in the original, and $\ell_{A,B}$ increments by one for each of these modifications.