Efficient Parallel Algorithms for Two Graph Layout Problems

Xin He [1]
Department of Computer Science
State University of New York at Buffalo
Buffalo, NY 14260

Abstract

We present efficient parallel algorithms for solving two graph layout problems: Find a Fáry Embedding on a grid and construct a rectangular dual for planar graphs. The algorithm for the first problem takes $O(\log n \log^* n)$ time with $O(n)$ processors on a PRAM. The algorithm for the second problem takes $O(\log^2 n)$ time with $O(n)$ processors.

1. Introduction

We present efficient parallel algorithms for solving two graph layout problems: Find a Fáry embedding on a grid and construct a rectangular dual for planar triangular graphs.

A Fáry embedding of a planar graph G is a plane embedding in which edges are represented by non-intersecting straight line segments joining their end vertices. It is known every planar graph has a Fáry embedding [5, 17]. Such embeddings can be useful in practice to visualize planar graphs on a graphic screen. Most initial Fáry embedding algorithms require high precision real arithmetic and the vertices tend to bunch together and thus making them useless in practice. In dealing with these drawbacks, Fraysseix, Pach and Pollack [6] developed an $O(n \log n)$ algorithm which computes a Fáry embedding on a $(n-2)\times(2n-4)$ grid. The running time of this algorithm was improved to $O(n)$ in [4]. Independently, Schnyder found an $O(n)$ time algorithm [15, 16] which computes a Fáry embedding on a $(n-2) \times (n-2)$ grid. Fürer and Raghavachari described a parallel implementation of Schnyder's algorithm using $O(\log^2 n)$ time with $O(n)$ processors [7]. We will present an $O(\log n \log^* n)$ time $O(n)$ processors parallel implementation of Schnyder's algorithm.

Given a rectangle R, a *subdivision system* of R is a subdivision of R into a set $\Phi = \{R_1, R_2, \ldots, R_n\}$ of non-intersecting smaller rectangles such that no four rectangles in Φ meet at the same point. A *rectangular dual* of a graph $G = (V, E)$ is a subdivision system Φ and a one-to-one correspondence $f : V \to \Phi$ such that two vertices $u, v \in V$ are adjacent in G if and only if the rectangles $f(u)$ and $f(v)$ share a common boundary (see Fig 4.1 and 4.2 for an example.) If G has a rectangular dual, clearly G must be planar. This problem finds applications in the floor planning of electronic chips and in

[1] Research supported by NSF grant CCR-9011214.

VLSI design. Linear time sequential algorithms for this problem were developed in [2, 3, 10, 13]. No parallel algorithm for this problem is known. We present an $O(\log^2 n)$ time $O(n)$ processors parallel algorithm for this problem. A more compact rectangular dual can be constructed in $O(\log^2 n)$ time with $M(n)$ processors, where $M(n)$ is the number of processors needed for multiplying two $n \times n$ matrices.

The parallel computation model used in this paper is a CRCW PRAM. The model consists of a number of processors. In each step, a processor can read from a memory cell, perform an arithmetic or a logic operation, and write into a memory cell. Both concurrent read and concurrent write are allowed. In case of a write conflict, an arbitrary processor succeeds.

Although the Fáry embedding and the rectangular dual problems appear quite different, the algorithmic tool for solving them are remarkably similar: both require to label the edges of the graph with certain regular properties. The present paper is organized as follows: In section 2, we define basic terms and introduce a lemma needed by both algorithms. Section 3 describes the Fáry embedding algorithm. Section 4 presents the rectangular dual algorithm.

2. Definitions

Let $G = (V, E)$ be a simple planar graph with n vertices and m edges. If $(v, u) \in E$, u is a *neighbor* of v. $deg(v)$ denotes the number of neighbors of a vertex v. A path (or a cycle) of G consisting of k edges is called a k-path (or a k-cycle, respectively). A *triangle* is a 3-cycle and a *quadrangle* is a 4-cycle.

Consider a fixed plane embedding of G. The embedding divides the plane into a number of *faces*. The unbounded face is the *exterior face*. Other faces are *interior faces*. The vertices and the edges on the boundary of the exterior face are called exterior vertices and exterior edges, respectively. A cycle C of G divides the plane into its interior and exterior region. If C contains at least one vertex in its interior, C is called a *separating cycle*.

A *planar triangular* graph is a planar graph in which all of its faces are triangles. For the two problems considered in this paper, as we will see later, we only need to consider planar triangular graphs. Let $G = (V, E)$ be such a graph. Let v be an interior vertex of G. Suppose $u_1, \ldots, u_k$ are the neighbors of v. We use $N(v)$ to denote the set of neighbors of v and $C(v)$ to denote the cycle consisting of $\{u_1, \ldots u_k\}$. The *star at* v, denoted by $Star(v)$, is the subgraph consisting of the edges $\{(v, u_i) \mid 1 \leq i \leq k\}$. Whenever the notations $N(v), C(v), Star(v)$ are used, it is understood that the members in each set are listed in counterclockwise order around v in the embedding.

We assume the embedding information of G is given by the following data structure. For each

$v \in V$, there is a doubly linked circular list $List(v)$ containing the vertices in $N(v)$ in counterclockwise order. The two copies of an edge (u, v) (one in $List(u)$ and one in $List(v)$) are cross-lined to each other. This data structure can be constructed as a by-product by using the planarity testing algorithm in $O(\log n)$ time with $O(n)$ processors [14].

Let I be an independent set of G (i.e. no two vertices in I are adjacent in G). I is a *large independent set* if $|I| \geq cn$ for some constant c. Let v be a vertex in I. A neighbor u of v is called a *bad neighbor* if u is also a neighbor of another vertex in I. Otherwise u is a *good neighbor* of v. In our algorithms, we wish to find a large independent set I such that for any $v \in I$, all neighbors of v are good. Unfortunately, this is not always possible. For any k, it's easy to construct a planar triangular graph consisting of k degree-4 vertices and two degree-k vertices such that every degree-4 vertex is adjacent to both degree-k vertices. It's also easy to construct a planar triangular graph consisting of $2k$ degree-5 vertices and two degree-k vertices such that half of the degree-5 vertices are adjacent to one degree-k vertex and another half of the degree-5 vertices are adjacent to another degree-k vertex. This motivates the following:

Definition 2.1: Let $G = (V, E)$ be a planar graph. A set $I \subset V$ is a *sparse independent set* (SIS), if it is independent and for all $v \in I$ the following hold: (1) $deg(v) \leq 6$; (2) If $deg(v) = 4$, then at least two neighbors of v are good; (3) If $deg(v) = 5$, then at least four neighbors of v are good; (4) If $deg(v) = 6$, then all neighbors of v are good.

The following construction finds a large SIS in any planar graph $G = (V, E)$ with minimum degree at least 3. (Note the minimum degree of any planar triangular graph is at least 3.) This construction was invented in [9] for constructing a 5-coloring of planar graphs.

Let $V_i = \{v \in V | deg(v) = i\}$ and $V_{[i,j]} = \{v \in V | i \leq deg(v) \leq j\}$. The vertices in $V_{[3,29]}$ are called *light* vertices. The vertices in $V_{[30,\infty]} = \{v \in V | 30 \leq deg(v)\}$ are called *heavy* vertices. Define: $U_4 = \{v \in V_4 | v$ has at least two light neighbors$\}$; $U_5 = \{v \in V_5 \mid v$ has at least four light neighbors$\}$; $U_6 = \{v \in V_6 \mid$ all neighbors of v are light $\}$; $U = V_3 \cup U_4 \cup U_5 \cup U_6$.

Let $G_1 = (V_{[3,29]}, E_1)$ be the subgraph of G induced by $V_{[3,29]}$. Let $G_1^s = (V_{[3,29]}, E_1^s)$ be the square graph of G_1. Namely, $(x, z) \in E_1^s$ if and only if there is a vertex y such that both $(x, y), (y, z) \in E_1$. Let $G_2 = (V, E \cup E_1^s)$ and G_3 be the subgraph of G_2 induced by the set U. Let I be a maximal independent set of G_3. The following lemma was proved in [9]:

Lemma 2.1: Let $G = (V, E)$ be a planar graph with minimum degree ≥ 3. Then (1) The set U defined above contains at least $n/24$ vertices. (2) The set I defined above is a large SIS of G, which can be constructed in $O(\log^* n)$ time with $O(n)$ processors.

Since the proof of (1) is non-trivial and not provided in [9], it is given in the appendix for completeness. To see I is an SIS of G, note that for any $v \in I$, if $deg(v) = 4$ then v has at least two light neighbors; if $deg(v) = 5$ then v has at least four light neighbors; if $deg(v) = 6$ then all neighbors of v are light. The construction guarantees that all light neighbors are good neighbors.

3. Fáry Embedding Algorithm

For Fáry embedding problem, we can assume G is a planar triangular graph since otherwise dummy edges can be added into G to make it triangular. After an embedding is obtained, the dummy edges can be removed. Section 3.1 reviews Schnyder's algorithm. Our implementation is discussed in section 3.2.

3.1 Schnyder's algorithm

Schnyder's algorithm is based on an elegant study on the "poset dimension" for planar graphs [15]. The presentation below has been tailored to facilitate parallel implementation.

Definition 3.1: A *normal edge labeling* (NEL) of a planar triangular graph G is a partition of the interior edges of G into three sets $\{T_1, T_2, T_3\}$ of directed edges such that the following hold:

(1) For each interior vertex v, the edges in $Star(v)$ appear around v in counterclockwise order as follows: one edge in T_1 leaving v; a set (maybe empty) of edges in T_3 entering v; one edge in T_2 leaving v; a set (maybe empty) of edges in T_1 entering v; one edge in T_3 leaving v; a set (maybe empty) of edges in T_2 entering v.

(2) Let v_1, v_2, v_3 be the three exterior vertices of G in counterclockwise order. All interior edges incident to v_i $(i = 1, 2, 3)$ are in T_i and entering v_i.

The NEL is called *realizer* in [15, 16]. The following theorem was proved in [15, 16]:

Theorem 3.1: Let G be a planar triangular graph with at least four vertices. Then G has a NEL $\{T_1, T_2, T_3\}$. Moreover, each T_i $(i = 1, 2, 3)$ is a tree including all interior vertices and exactly one exterior vertex v_i and all edges of T_i are directed toward v_i.

Let $\{T_1, T_2, T_3\}$ be an NEL of $G = (V, E)$. For each $v \in V$, let $P_i(v)$ $(i = 1, 2, 3)$ denote the path in T_i from v to the root of T_i. It was shown in [15, 16] that v is the only common vertex of the paths $P_i(v)$ and $P_j(v)$ $(i \neq j)$. So $P_1(v), P_2(v), P_3(v)$ divides G into three regions $R_1(v), R_2(v), R_3(v)$, where $R_i(v)$ denotes the closed region opposite to the root of T_i. Define:

$p_i(v) = |P_i(v)|$, the number of vertices on the path $P_i(v)$.

$r_i(v) = |R_i(v)|$, the number of vertices in the closed region $R_i(v)$.

For the three exterior vertices v_i $(i = 1, 2, 3)$ define: $p_i(v_i) = 1$, $p_j(v_i) = 2$ for $j \neq i$; and $r_i(v_i) = n$, $r_j(v_i) = 2$ for $j \neq i$. For each $v \in I$, define:

$x(v) = r_3(v) - p_2(v) =$ the number of vertices in the region $R_3(v)$ but not on the path $P_2(v)$;

$y(v) = r_1(v) - p_3(v) =$ the number of vertices in the region $R_1(v)$ but not on the path $P_3(v)$.

The following theorem is the basis of Schnyder's algorithm.

Theorem 3.2 [15, 16]: Let $\{T_1, T_2, T_3\}$ be an NEL of a planar triangular graph $G = (V, E)$. Then the assignment of each vertex $v \in V$ to the grid point $(x(v), y(v))$, where $x(v)$ and $y(v)$ are as defined above, induces a Fáry embedding of G on a $(n-2) \times (n-2)$ grid.

Fig 1

Fig 1 shows an example of this construction. (In this example, $p_1(a) = 2, p_2(a) = 3, p_3(a) = 3, r_1(a) = 6, r_2(a) = 5, r_3(a) = 4$, etc.) Schnyder's algorithm is as follows [16]:

Algorithm 1: Fáry embedding of a planar triangular graph $G = (V, E)$.

(1) Construct an NEL $\{T_1, T_2, T_3\}$ of G.

(2) For each $v \in V$, compute $p_i(v), r_i(v)$ $(i = 1, 2, 3)$.

(3) For each $v \in V$, embed v at the grid point $(x(v), y(v))$ as in Theorem 3.2.

End Algorithm 1.

Let $t_i(v)$ be the number of vertices in the subtree of T_i rooted at v. The quantity $r_i(v)$ can be computed by using the formula [15, 16, 7]: $r_i(v) = \sum_{x \in P_{i+1}(v)} t_i(x) + \sum_{x \in P_{i-1}} t_i(x) - t_i(v)$.

After an NEL of G has been found, $p_i(v)$ and $t_i(v)$ for each $v \in V$ can be computed in parallel by using the tree contraction method [1] on T_i. Then $r_i(v)$ can be computed by another application of tree contraction on T_i using above formula. All these calculations take $O(\log n)$ time with $O(n/\log n)$ processors [7]. So the central part of the algorithm is step (1) which will be discussed in section 3.2.

3.2 Constructing an NEL for a planar triangular graph

The basic method for constructing an NEL of G is the *edge contraction* and *edge expansion* [16]. In order to find an NEL of G in parallel, we follow the approach used in [7]: contract a large set E_1 from G simultaneously. Our algorithm differs from [7] in the way the set E_1 is constructed. We begin with the definition of contractable edges.

Let v be an interior vertex of G and u a neighbor of v. Let y and z be the two neighbors of v that are consecutive with u in $C(v)$. If u is **not** adjacent to any neighbor of v other than y and z, we say u is a *contractable neighbor* of v and (v, u) a *contractable edge*. Note that (v, u) is a contractable edge if and only if (v, u) is not on any separating triangle of G. By the planarity, it's easy to show

every interior vertex has at least two contractable neighbors.

Let $e = (v, u)$ be a contractable edge of G. The operation *contracting* e identifies v with u. More precisely, the operation removes v and the edges in $Star(v)$ from G, then add a new edge (x, u) for each neighbor x of v not consecutive with u. The resulting graph is called the *contracted graph* and denoted by G/e. Note G/e is a planar triangular graph.

Let $\{T_1', T_2', T_3'\}$ be an NEL of $G' = G/e$. The operation *expanding* e undoes the operation *contracting* e and obtain an NEL of G. Fig 2 shows two examples of edge expansions of several possible cases. Other cases can be found in [7]. (The figure only shows the labels of relevant edges.)

Fig 2

Two contractable edges e_1 and e_2 are *compatible* (as defined in [7]) if and only if: (a) they do not share a common vertex; and (b) they do not form opposite sides of a quadrangle with other two edges of G. Let E_1 be a set of pairwise compatible contractable edges. It was shown in [7] that all edges in E_1 can be contracted from G and then expanded simultaneously without interfering with each other. We use G/E_1 to denote the graph obtained from G by contracting all edges in E_1 simultaneously.

In order to find a large set E_1 of compatible contractable edges, we first find a large SIS I as in Lemma 2.1. For each vertex $v \in I$, we select a contractable neighbor u for v. For a degree-3 vertex $v \in I$, every neighbor u of v is contractable. The effect of contracting (v, u) is equivalent to deleting v from G. So we only need to consider the cases $deg(v) = 4, 5, 6$.

Case 1: $deg(v) = 6$. Select any contractable neighbor u of v (which must be good).

Case 2: $deg(v) = 5$. Select a good contractable neighbor u of v. (Since v has at least two contractable neighbors and at most one bad neighbor, such a vertex u always exists.)

Case 3: $deg(v) = 4$. If v has a good contractable neighbor u, then select u. If v has no good contractable neighbor, the vertices in $C(v) = \{x, w, y, u\}$ (in counterclockwise order) must satisfy the following conditions: u and w are good and are adjacent in G; x and y are bad and are contractable (Fig 3.1). The triangle $C = \{u, v, w\}$ contains exactly one of x and y in its interior. If x is in the interior of C, then select x. If y is in the interior of C, then select y.

Fig 3

Define: $E_1 = \{(v, u) \mid v \in I,\ 4 \leq deg(v) \leq 6,\ u$ is the neighbor of v selected as above$\}$.

Lemma 3.3: E_1 is a set of compatible contractable edges of G.

Proof: Since we always select a contractable neighbor for each v, every edge in E_1 is contractable. In order to show E_1 is compatible, consider any two edges $e_1 = (v_1, y_1)$ and $e_2 = (v_2, y_2)$ in E_1. If

$deg(v_1) = 5$ or 6, then y_1 is a good neighbor of v_1 and hence cannot be a neighbor of v_2. Thus e_1 and e_2 cannot share a common vertex, nor can they form opposite sides of a quadrangle. If $deg(v_2) = 5$ or 6, the same claims can be made. So we only need to consider the case $deg(v_1) = deg(v_2) = 4$.

Suppose $e_1 = (v_1, y_1), e_2 = (v_2, y_2) \in E_1$ share a common vertex $y_1 = y_2$. Since y_1 is selected for v_1, the vertices in $C(v_1) = \{x_1, w_1, y_1, u_1\}$ must satisfy: x_1 and y_1 are bad; u_1 and w_1 are good and are adjacent in G; y_1 is in the interior of the triangle $\{u_1, v_1, w_1\}$ (Fig 3.2). Since y_2 is selected for v_2, the vertices in $C(v_2) = \{x_2, w_2, y_2, u_2\}$ must satisfy: x_2 and y_2 are bad; u_2 and w_2 are good and are adjacent in G; y_2 is in the interior of the triangle $\{u_2, v_2, w_2\}$. This is impossible by the planarity.

Suppose e_1 and e_2 form opposite sides of a quadrangle. The situation must be the one shown in Fig 3.3. Again, this is impossible by the planarity. □

The following is our algorithm:

Algorithm 2: Find an NEL of a planar triangular graph $G = (V, E)$.

(1) If G has only one interior vertex, return the trivial NEL and stop.

(2) Find a sparse independent set I of G as in Lemma 2.1.

(3) For each $v \in I$ with degree 4, 5, or 6, select a contractable neighbor of v as described above. Let E_1 be the set of selected contractable edges.

(4) Remove all degree-3 vertices in I from G. Contract all edges in E_1 from G. Call the resulting graph G_1.

(5) Recurse on G_1 to find an NEL of G_1.

(6) Expand the NEL of G_1 to be an NEL of G.

End Algorithm 2.

The correctness of the algorithm follows from above discussion. We show Algorithm 2 can be implemented in $O(\log n \log^* n)$ time with $O(n)$ processors. Before the execution, G is pre-processed so that $deg(v)$ is computed for each v. The pre-processing takes $O(\log n)$ time with $O(n)$ processors. Since each iteration of Algorithm 2 removes $|I| \geq cn$ vertices for a fixed constant c, the recursion depth is $O(\log n)$. So it's enough to show each iteration takes $O(\log^* n)$ time with $O(n)$ processors.

Step (1) clearly takes $O(1)$ time with one processor. Step (2) takes $O(\log^* n)$ with $O(n)$ processors by Lemma 2.1. To implement Step (3), we need to select a contractable neighbor u for each $v \in I$ with $4 \leq deg(v) \leq 6$. The selection of u is based on two criteria: (1) Whether u is a good neighbor

or not. (2) If u is good, whether it is adjacent to a non-consecutive neighbor of v or not. Since $deg(v) \leq 6$ and all good neighbors of v have bounded degree (being light vertices), the selection can be determined by looking at $deg(u)$ and by scanning the adjacency list of u in $O(1)$ time. Thus the selection of E_1 takes $O(1)$ time with $O(n)$ processors.

Step (4): To contract an edge $(v, u) \in E_1$ (or delete a degree-3 vertex $v \in I$) from G, we need to delete v and modify the adjacency lists and the degrees of v's neighbors. Since $deg(v) \leq 6$, this can be done in $O(1)$ time by one processor per edge contraction. Hence the entire step takes $O(1)$ time with $O(n)$ processors.

Step (5) is recursive call. The expansion step (6) can be done in $O(1)$ time with $O(n)$ processors. This completes the analysis of the algorithm. In summary we have:

Theorem 3.4: A Fáry embedding of a planar triangular graph G on an $(n-2) \times (n-2)$ grid can be constructed in $O(\log n \log^* n)$ time with $O(n)$ processors on a PRAM.

4. Rectangular Dual Algorithm

In section 4.1, we review the sequential rectangular dual algorithm in [10]. The central part of the algorithm is to find a *regular edge labeling*. Sections 4.2 and 4.3 are devoted to the techniques for constructing such a labeling. Section 4.4 presents the algorithm.

4.1 Sequential algorithm

Consider a planar graph H. Let v_0, v_1, v_2, v_3 be four distinct vertices on the exterior face of H in counterclockwise order. Let P_i $(i = 0, 1, 2, 3)$ be the path on the exterior face of H consisting of the vertices between v_i and v_{i+1} (where addition is mod 4). We seek a rectangular dual R_H of H such that the four vertices v_0, v_1, v_2, v_3 correspond to the four corner rectangles of R_H and the vertices on P_0 (P_1, P_2, P_3, respectively) correspond to the rectangles located on the north (west, south, east, respectively) boundary of R_H. In order to simplify the problem, we modify H as follows: Add four new vertices v_N, v_W, v_S, v_E and connect v_N (v_W, v_S, v_E, respectively) to every vertex on P_0 (P_1, P_2, P_3, respectively). Then add four new edges (v_S, v_W), (v_W, v_N), (v_N, v_E), (v_E, v_S). Let G be the resulting graph. It's easy to see that H has a rectangular dual R_H if and only if G has a rectangular dual R with exactly four rectangles on the boundary of R. The following theorem is proved in [13, 10]:

Theorem 4.1: Let G be a planar graph. G has a rectangular dual R with four rectangles on the boundary of R if and only if the following hold: (1) Every interior face of G is a triangle and the exterior face of G is a quadrangle. (2) G has no separating triangles.

A graph G satisfying the two conditions in Theorem 4.1 is called a *proper triangular planar* (PTP) graph. From now on, we will only discuss such graphs. Note that since G has no separating triangles, the minimum degree of G is at least 4.

Definition 4.1: A *regular edge labeling* (REL) of a PTP graph G is a partition of the interior edges of G into two subsets $\{T_1, T_2\}$ of directed edges such that the following hold:

(1) For each interior vertex v, the edges in $Star(v)$ appear in counterclockwise order around v as follows: a set of edges in T_1 leaving v; a set of edges in T_2 entering v; a set of edges in T_1 entering v; a set of edges in T_2 leaving v.

(2) All interior edges incident to v_N (v_S, respectively) are in T_1 and directed entering v_N (leaving v_S, respectively). All interior edges incident to v_E (v_W, respectively) are in T_2 and directed entering v_E (leaving v_W, respectively).

It was show in [10] that every PTP graph G has a REL $\{T_1, T_2\}$. Let G_1 be the directed subgraph of G induced by the edges in T_1 and the four exterior edges directed as: $v_S \rightarrow v_W$, $v_W \rightarrow v_N$, $v_S \rightarrow v_E$, $v_E \rightarrow v_N$. It can be shown G_1 is acyclic with v_N as the only sink and v_S as the only source. For each edge $e \in T_1$, let $left(e)$ ($right(e)$, respectively) be the face of G_1 on the left (right, respectively) of e.

The *dual graph* of G_1, denoted by G_1^*, is defined as follows. The node set of G_1^* is the set of interior faces of G_1. For each edge $e \in T_1$, there is a corresponding arc e^* in G_1^* directed from the face $left(e)$ to the face $right(e)$. Let w^* and e^* be the faces adjacent to the vertices v_W and v_E respectively. G_1^* is acyclic with w^* as the only source and e^* as the only sink.

A *consistent numbering* of G_1^* *of order* X is a surjective mapping F_1 from the set of nodes in G_1^* to the set of integers $\{1, \ldots, X\}$ such that (a) $F_1(w^*) = 1$, $F_1(e^*) = X$; and (b) If there is an arc from the node f^* to the node g^* in G_1^*, then $F_1(f^*) < F_1(g^*)$. For an example, a topological ordering [12] of G_1^* is a consistent numbering of G_1^*, which will be denoted by $top_1(f^*)$. As another example, if we define $level_1(f^*)$ to be the length of the longest path from w^* to f^* in G_1^*, then $level_1$ is also a consistent numbering of G_1^*. As we will see later, $level_1$ produces a more compact rectangular dual than top_1 does. However, $level_1$ requires $M(n)$ processors to compute while top_1 only requires $O(n)$ processors to compute.

Let F_1 be a consistent numbering of G_1^* of order X. For each interior vertex v, let e_1 and e_2 be the left-most and the right-most edge entering v in G_1. Define $x_1(v) = F_1(left(e_1))$ and $x_2(v) = F_1(right(e_2))$. For the four exterior vertices, define $x_1(v_W) = 0$, $x_2(v_W) = 1$; $x_1(v_E) = X$, $x_2(v_W) = X + 1$; $x_1(v_S) = x_1(v_N) = 1$, $x_2(v_S) = x_2(v_N) = X$.

Let G_2 be the directed subgraph of G induced by the edges in T_2 and the four exterior edges directed as: $v_W \rightarrow v_S$, $v_S \rightarrow v_E$, $v_W \rightarrow v_N$, $v_N \rightarrow v_E$. It can be shown G_2 is acyclic with v_E as the only sink and v_W as the only source. For each edge $e \in T_2$, let $above(e)$ ($below(e)$, respectively) be the face of G_2 on the left (right, respectively) of e.

The *dual graph* of G_2, denoted by G_2^*, is defined as follows: The node set of G_2^* is the set of interior faces of G_2. For each edge $e \in T_2$, there is a corresponding arc e^* in G_2^* directed from the face $below(e)$ to the face $above(e)$. Let s^* and n^* be the faces adjacent to the vertices v_S and v_N, respectively. G_2^* is acyclic with s^* as the only source and n^* as the only sink. A consistent numbering of G_2^* can be defined similarly.

Let F_2 be a consistent numbering of G_2^* of order Y. For each interior vertex v, let e_3 and e_4 be the right-most and the left-most edge entering v in G_2. Define $y_1(v) = F_2(below(e_3))$ and $y_2(v) = F_2(above(e_4))$. For the four exterior vertices, define $y_1(v_W) = y_1(v_E) = 0$; $y_2(v_W) = y_2(v_E) = Y + 1$; $y_1(v_S) = 0$; $y_2(v_S) = 1$; $y_1(v_N) = Y$, $y_2(v_N) = Y + 1$.

The following theorem is the basis of the rectangular dual algorithm.

Theorem 4.2 [10]: Let $\{T_1, T_2\}$ be a REL of a PTP graph $G = (V, E)$. Let F_1 and F_2 be two consistent numbering of G_1^* and G_2^* of order X and Y, respectively. For each $v \in V$, let $R(v)$ be the rectangle bounded by the four lines: $x = x_1(v), x = x_2(v), y = y_1(v), y = y_2(v)$, where $x_1(v), x_2(v), y_1(v), y_2(v)$ are as defined above. Then the set $\{R(v) \mid v \in V\}$ forms an $(X+1) \times (Y+1)$ rectangular dual of G.

Fig 4

Fig 4 shows an example of this construction with $F_1 = level_1$ and $F_2 = level_2$. ($level_2(g^*)$ is the length of the longest path in G_2^* from s^* to g^*.) Fig 4.1 shows a PTP graph G. Fig 4.3 is the subgraph G_1. The integers in small boxes are the $level_1$-numbers of the faces of G_1. Fig 4.4 is the subgraph G_2. The integers in small boxes are the $level_2$-numbers of the faces of G_2. Fig 4.2 shows the rectangular dual of G constructed. Our rectangular dual algorithm is as follows:

Algorithm 3: Rectangular Dual of a PTP graph $G = (V, E)$.

(1) Find a REL $\{T_1, T_2\}$ of G.

(2) Construct the graph G_1 and its dual graph G_1^*. Compute a consistent numbering F_1 of G_1^*. Compute $x_1(v), x_2(v)$ for each $v \in V$.

(3) Construct the graph G_2 and its dual graph G_2^*. Compute a consistent numbering F_2 of G_2^*. Compute $y_1(v), y_2(v)$ for each $v \in V$.

(4) Assign each $v \in V$ a rectangle $R(v)$ bounded by the coordinates $x_1(v), x_2(v), y_1(v), y_2(v)$.

End Algorithm 3.

To implement this algorithm in parallel, note that the graph G_1 and G_1^* can be constructed in $O(\log n)$ time with $O(n)$ processors using the embedding information. The implementation of Step (2) depends on the choice of F_1. If we choose the topological ordering top_1, this step takes $O(\log^2 n)$ time with $O(n)$ processors by using the algorithm in [12]. If we choose $level_1$, this step can be implemented by using the breadth-first search in G_1^* in $O(\log^2 n)$ time with $M(n)$ processors [8]. (A more compact rectangular dual will be computed in this case). $x_1(v), x_2(v)$ for each $v \in V$ can be computed in $O(1)$ time with $O(n)$ processors. The implementation of Step (3) is similar. Step (4) takes $O(1)$ time with $O(n)$ processors. Thus the central part of Algorithm 3 is Step (1). This will be discussed in the following subsections.

4.2 Compatible contractable edges

As in Section 3, the basic method for constructing a REL of G is edge contraction and expansion. We begin with the definition of the contractable edges. Although we use the same term as in Section 3, it has different meaning in this section.

Consider a vertex $v \in V$ and a neighbor u of v. Let y and z be the two neighbors of v that are consecutive with u in $C(v)$. Since G has no separating triangles, u is not adjacent to any neighbor of v other than y and z. If u is **not** connected by a 2-path to any neighbor of v other than y and z, u is a *contractable neighbor* of v and (v, u) is a *contractable edge*. Note that an edge e is contractable if and only if e is not on any separating quadrangle of G.

The *contracted graph* $G/(v, u)$ by a contractable edge (v, u) is defined as in section 3.2. Since (v, u) is not on any separating quadrangle of G, the graph $G' = G/(v, u)$ has no separating triangles. So G' is a PTP graph. If a REL $\{T_1', T_2'\}$ of G' has been found, we can expand it to be a REL $\{T_1, T_2\}$ of G. As we will see later, we only need to consider the cases $deg(v) = 4, 5$, or 6.

The case $deg(v) = 4$ is simple. Suppose $N(v) = \{u, w, x, y\}$ in counterclockwise order. To contract $e = (v, u)$, we delete v and the edges in $Star(v)$ and add a new edge (x, u). The expansion is completely determined by the the label of (x, u) in $\{T_1', T_2'\}$. An example is shown in Fig 5.1. Suppose $(x, u) \in T_2'$ and is directed as $x \rightarrow u$. Then after the expansion, the edges in $Star(v)$ are labeled as follows: $(x, v), (v, u) \in T_2$ and directed as $x \rightarrow v, v \rightarrow u$; $(y, v), (v, w) \in T_1$ and directed as $y \rightarrow v, v \rightarrow w$. Note that for any edge e' on $C(v)$ or in the exterior of $C(v)$, the label of e' in $\{T_1', T_2'\}$ is the same as it's label in $\{T_1, T_2\}$. This is also true for other cases.

If $deg(v) = 5$ or 6, the situation is more complex. We only consider the cases $deg(v) = 6$. (The case $deg(v) = 5$ is similar). Suppose $C(v) = \{t, u, w, x, y, z\}$ in counterclockwise order. To contract

(v,u), we delete v and the edges in $Star(v)$, then add edges $(x,u),(y,u),(z,u)$. Depending on the labels of the edges $(t,u),(w,u),(x,u),(y,u),(z,u)$ in $\{T_1', T_2'\}$, there are a number of cases. Three of them are shown in Fig 5.2 through 5.4 (only the labels of relevant edges are shown). Note that for any edge e', if e' is in the exterior of $C(v)$ or $e \in C(v) - \{(t,u),(w,u)\}$, then the label of e' in $\{T_1', T_2'\}$ is the same as its label in $\{T_1, T_2\}$. This is also true for other cases. However, the labels of (t,u) and (w,u) in $\{T_1', T_2'\}$ may differ from their labels in $\{T_1, T_2\}$ (e.g. Fig 5.3 and 5.4). This situation complicates the algorithm. We call (t,u) and (w,u) the *nasty edges* of (v,u).

Fig 5

In order to construct a REL of G in parallel, we need to find a set of edges that can be contracted from G simultaneously.

Definition 4.2: A set E_1 of contractable edges of G is *compatible* if the following hold:

(1) No two edges in E_1 share a common vertex;

(2) No two edges in E_1 form opposite sides of a quadrangle;

(3) No two edges in E_1 are on a separating 5-cycle;

(4) No three edges in E_1 are on a separating 6-cycle.

Lemma 4.3: Let E_1 be a set of compatible contractable edges of a PTP graph G. Then all edges in E_1 can be contracted from G simultaneously so that the resulting graph is a PTP graph.

Proof: By induction on $|E_1|$. If $|E_1| = 1$, there is nothing to be proved. Suppose $|E_1| > 1$. Consider an edge $e = (v,u) \in E_1$. Let $G' = G/e$ and O be the vertex in G' corresponding to the contracted edge e. We show $E' = E_1 - \{e\}$ is a set of compatible contractable edges with respect to G'. For any $e' \in E'$, since e' and e are not on any separating 5-cycle of G, e' is not on any separating quadrangle of G'. Hence every edge in E' is a contractable edge of G'. Next we show E' satisfies all conditions of Definition 4.2 with respect to G'.

(1) Suppose two edges $e_1, e_2 \in E'$ share a common vertex in G'. Since e_1, e_2 share no vertex in G, the common vertex must be the "shrunken" vertex O. So both e_1 and e_2 have a common vertex with e in G. This contradicts the fact that E_1 is compatible with respect to G.

(2) Suppose e_1 and e_2 form opposite sides of a quadrangle Q of G'. The shrunken vertex O must be on Q. This implies that either e_1 or e_2 share a common vertex with e in G. This contradicts the fact that E_1 is compatible with respect to G.

(3) Since no three edges in E_1 are on any separating 6-cycle in G, no two edges in E' are on a separating 5-cycle in G'.

(4) Similar to (1) and (2), it can be shown no three edges in E' are on a separating 6-cycle of G'.

Thus E' is a set of compatible contractable edges of G'. By induction hypothesis, E' can be contracted from G' simultaneously. So E_1 can be contracted from G simultaneously. □

4.3 Find a large set of compatible contractable edges

The basic idea for finding a REL of a PTP graph $G = (V, E)$ is similar to the method used in section 3.2: Find a set E' of compatible contractable edges; obtain the contracted graph $G' = G/E'$; recurse on G' to find a REL $\{T_1', T_2'\}$ of G'; finally expand the REL of G' to be a REL $\{T_1, T_2\}$ of G.

In order to find E', we use the large SIS I of G as in Lemma 2.1. Since G has no degree-3 vertices, $I = I_4 \cup I_5 \cup I_6$, where $I_j = \{v \in I \mid deg(v) = j\}$ for $j = 4, 5, 6$. For each $v \in I_j$, we try to select a contractable neighbor u of v and (tentatively) define:

$E_j = \{(v, u) \mid v \in I_j$ and u (if it exists) is a selected contractable neighbor of $v\}$.

We wish to show each E_j can be contracted from G and then expanded simultaneously. Unfortunately, this is not always possible for the following three reasons:

(a) Some $v \in I_6$ might have no contractable neighbors and some $v \in I_5$ might only have a bad contractable neighbor.

(b) E_5 and E_4 need not be compatible because the condition (3) of Definition 4.2 might fail.

(c) Even if E_j is compatible for G, Lemma 4.3 only states that E_j can be contracted from G simultaneously so that the resulting graph is a PTP graph. However, two edges $e_1 = (v_1, u_1), e_2 = (v_2, u_2) \in E_j$ may interfere with each other during the expansion stage. We examine below how and where this expansion interference can occur.

Consider E_4. For any edge e' on $C(v_1)$ or in the exterior of $C(v_1)$, the label of e' in $\{T_1', T_2'\}$ is the same as its label in $\{T_1, T_2\}$. This is also true for the edges in the exterior of or on $C(v_2)$. So when expanding e_1 and e_2, the labels of the edges in $Star(v_1)$ and $Star(v_2)$ can be determined independently. Therefore the edges in E_4 can be expanded simultaneously.

Consider E_5 and E_6. Recall that for any edge e' in the exterior of $C(v_1)$ or on $C(v_1)$ but not the two nasty edges of (v_1, u_2), the label of e' in $\{T_1', T_2'\}$ is the same as its label in $\{T_1, T_2\}$. However, the labels of the two nasty edges of (v_1, u_1) in $\{T_1', T_2'\}$ may differ from their labels in $\{T_1, T_2\}$. If a nasty edge of (v_1, u_1) and a nasty edge of (v_2, u_2) are on a triangle face of G, then e_1 and e_2 might interfere with each other during expansion.

Consider two edges $e_1, e_2 \in E_6$. Since all neighbors of v_1 and v_2 are good, $C(v_1)$ and $C(v_2)$ are disjoint. So the nasty edges of (v_1, u_1) and the nasty edges of (v_2, u_2) cannot be on a triangle face of G. So E_6 can be expanded simultaneously.

Unfortunately, the expansion interference can happen to E_5. We next discuss details on how to

select contractable neighbors and how to overcome these difficulties.

Case 1: $v \in I_6$. If v has a contractable neighbor u (which must be good), then select u. If v has no contactable neighbors, then each $u \in C(v)$ is connected by a 2-path to another neighbor of v that is not consecutive with u. By the planarity and the fact that G has no separating triangle, the situation must be the one shown in Fig 6.1. Note that: $deg(y) = deg(z) = 4$; $deg(x) \geq 5$; $deg(t) \geq 5$; $deg(p) \geq 5$. (If $deg(p) = 4$, (x, t) must be an edge of G. Then $\{t, v, x\}$ form a separating triangle. This contradicts that G is a PTP graph.)

Define: $I_6' = \{v \in I_6 \mid v$ has a contractable neighbor $\}$;

$E_6 = \{(v, u) \mid v \in I_6'$ and u is a contractable neighbor of $v\}$;

$I_6'' = \{y \mid y$ is a degree-4 neighbor of a vertex $v \in I_6 - I_6'$ as shown in Fig 6.1 $\}$.

Fig 6

Case 2: $v \in I_5$. By the planarity and the assumption that G has no separating triangles, it can be shown v has at least one contractable neighbor. If v has a good contractable neighbor u, then select u. If v has no good contractable neighbor, then each good neighbor u of v is connected by a 2-path to another neighbor of v that is not consecutive with u. The situation must be the one shown in Fig 6.2 (where w is the bad neighbor of v). Note that: $deg(y) = deg(z) = 4$; $deg(x) \geq 5$; $deg(t) \geq 5$; $deg(p) \geq 5$. (If $deg(x) = 4$, then (p, w) must be an edge of G and $\{p, w, t\}$ is a separating triangle. This contradicts that G is a PTP graph. The proof of other two inequalities is similar.)

Define: $I_5' = \{v \in I_5 \mid v$ has a good contractable neighbor $\}$;

$E_5 = \{(v, u) \mid v \in I_5'$ and u is a good contractable neighbor of $v\}$;

$I_5'' = \{y \mid y$ is a degree-4 neighbor of a vertex $v \in I_5 - I_5'$ as shown in Fig 6.2 $\}$.

Before we discuss the degree-4 vertices, we first prove the following:

Lemma 4.4: The set $I_4' = I_4 \cup I_5'' \cup I_6''$ is a sparse independent set of G.

Proof: By construction, every vertex in I_4' has degree 4. We need to show if $y \in I_4'$ then y is not adjacent to other vertices in I_4' and y has at least two good neighbors. Suppose $y \in I_6''$. From Fig 6.1, v and z cannot be in I_4'. Since x and p have degree at least 5, they cannot be in I_4' either. So y is not adjacent to other vertices in I_4'. Since v and z cannot be the neighbor of another vertex in I_4', they are two good neighbors of y. If $v \in I_5''$, the proof is similar. □

Case 3: $v \in I_4'$. By the planarity and the assumption that G has no separating triangles, it can be shown v has at least two contractable neighbors. If v has a good contractable neighbor u, then select u. If v has no good contractable neighbors, the situation must be the one shown in Fig. 6.3: x and y are contractable bad neighbors; u and w are good neighbors and are connected by a 2-path

$P = \{u, z, w\}$. Let Q be the quadrangle consisting of $\{u, v, w, z\}$. If x is in the interior of Q, then select x. If y is in the interior of Q, then select y. (There might be another 2-path $P' = \{u, z', w\}$ such that x is in the interior of Q and y is in the interior of the quadrangle $Q' = \{u, v, w, z'\}$. In this case, arbitrarily select either x or y.) Define:

$E_4 = \{(v, u) \mid v \in I'_4$ and u is a contractable neighbor of v selected as above$\}$.

Note that $|E_4| + |E_5| + |E_6| = |I'_4| + |I'_5| + |I'_6| = |I_4| + |I_5| + |I_6| = |I|$. We next discuss the compatibility of the sets E_6, E_5, E_4.

Lemma 4.5: E_6 is a compatible set.

Proof: Let $e_1 = (v_1, u_1)$ and $e_2 = (v_2, u_2)$ be any two edges in E_6. If any condition of Definition 4.2 fails, then v_1 and v_2 must have a common neighbor. Since all neighbors of v_1 and v_2 are good, this is impossible. Thus E_6 is compatible. □

Lemma 4.6: E_5 satisfies the conditions (1), (2) and (4) of Definition 4.2.

Proof: Let $e_1 = (v_1, u_1)$ and $e_2 = (v_2, u_2)$ be any two edges in E_5. If any of the conditions (1), (2) and (4) of Definition 4.2 fails, then either u_1 or u_2 (or both) must be a common neighbor of v_1 and v_2. Since we only select good neighbors for v_1 and v_2, this is impossible. □

E_5, however, needs not be compatible since the condition (3) of Definition 4.2 may fail. Moreover, two edges $e_1 = (v_1, u_1)$ and $e_2 = (v_2, u_2)$ in E_5 may interfere with each other during expansion. As mentioned earlier, this can happen only if a nasty edge of (v_1, u_1) and a nasty edge of (v_2, u_2) are on a triangle face of G. We will delete some edges from E_5 so that the resulting set $\overline{E}_5$ is compatible and no two edges in $\overline{E}_5$ interfere during expansion. In order to do so, we construct a *conflict graph* $\overline{D}$. The node set of $\overline{D}$ is the edges in E_5. Two nodes e_1, e_2 are adjacent in $\overline{D}$ if and only if either they are on a separating 5-cycle of G or they interfere with each other during expansion.

Define $E_D = \{C(v) \mid (v, u) \in E_5\}$. Let $D = (V_D, E_D)$ be the subgraph of G induced by the edge set E_D. Each connected component of D is a collection of $C(v)$'s sharing a common neighbor x. This connected component is denoted by $flower(x)$. Each $C(v)$ in $flower(x)$ is called a *petal*. The four good neighbors of v in $C(v)$ are called *tip vertices* of the petal. D is a collection of disjoint flowers. If $C(v)$ share no vertices with other cycles, it is considered a flower with one petal.

Consider a $flower(x)$ with t petals $C(v_1), \ldots, C(v_t)$ around x in counterclockwise order. Construct a graph $\overline{X}$ for $flower(x)$ as follows. $\overline{X}$ contains a t-cycle consisting of the nodes $\overline{v}_i$ $(i = 1 \ldots t)$. The node $\overline{v}_i$ corresponds to the pedal $C(v_i)$. In addition, for any two petals $C(v_i)$ and $C(v_j)$ of $flower(x)$, if a tip vertex of $C(v_i)$ is adjacent to a tip vertex of $C(v_j)$ in G, we add an edge $(\overline{v}_i, \overline{v}_j)$ in $\overline{X}$. It is easy to see $\overline{X}$ is an outer-planar graph. The conflict graph $\overline{D}$ is the union of $\overline{X}$'s obtained

from all flowers of D. (For a flower containing only one petal $C(v)$, there is an isolate node $\overline{v}$ in $\overline{D}$.) So $\overline{D}$ is a planar graph. Let $\overline{J}$ be an independent set in $\overline{D}$. Define:

$\overline{E}_5 = \{(v,u) \in E_5 \mid$ the node $\overline{v}$ in $\overline{D}$ that corresponds to $C(v)$ is in $\overline{J}\}$.

Lemma 4.7 $\overline{E}_5$ is compatible and no two edges in $\overline{E}_5$ interfere with each other during expansion.

Proof: $\overline{E}_5$ is obtained from E_5 by deleting some edges. Since E_5 satisfies the conditions (1), (2) and (4) of Definition 4.2, so does $\overline{E}_5$. We show it also satisfies the condition (3).

If two edges $e_i = (v_i, u_i), e_j = (v_j, u_j) \in E_5$ are on a separating 5-cycle of G, then v_i and v_j share a common neighbor x and (u_i, u_j) is an edge in G. Hence $C(v_i)$ and $C(v_j)$ belong to $flower(x)$ in D, and the nodes $\overline{v}_i$ and $\overline{v}_j$ are adjacent in $\overline{D}$. Since $\overline{J}$ is independent, at most one of e_1 and e_2 is in $\overline{E}_5$. So no two edges in $\overline{E}_5$ are on a separating 5-cycle of G.

If e_i and e_j interfere during expansion, then $C(v_i)$ and $C(v_j)$ must share a common vertex x and $C(v_i)$, $C(v_j)$ must be two consecutive petals of $flower(x)$ in D. Hence the nodes $\overline{v}_i$ and $\overline{v}_j$ are adjacent in $\overline{D}$. So at most one of e_1 and e_2 is in $\overline{E}_5$. Hence no two edges in $\overline{E}_5$ interfere with each other during expansion. □

Lemma 4.8: The set E_4 satisfies the conditions (1), (2) and (4) of Definition 4.2.

Proof: Consider two edges $e_1 = (v_1, y_1)$ and $e_2 = (v_2, y_2)$ in E_4. Suppose e_1 and e_2 share a common vertex $y_1 = y_2$. Since y_1 is selected for v_1, the vertices in $C(v_1) = \{y_1, u_1, x_1, w_1\}$ must satisfy: y_1 and x_1 are bad neighbors; u_1 and w_1 are good neighbors that are connected by a 2-path consisting of $\{u_1, z_1, w_1\}$; y_1 is in the interior of the quadrangle $\{u_1, z_1, w_1, v_1\}$ (Fig 7.1). Since y_2 is selected for v_2, the vertices in $C(v_2) = \{y_2, u_2, x_2, w_2\}$ must satisfy: y_2 and x_2 are bad neighbors; u_2 and w_2 are good neighbors that are connected by a 2-path consisting of $\{u_2, z_2, w_2\}$; y_2 is in the interior of the quadrangle $\{u_2, z_2, w_2, v_2\}$. By the planarity and the assumption that G has no separating triangles, this is impossible.

If e_1 and e_2 form opposite sides of a quadrangle, the situation must be the one shown in Fig 7.2. Again, this is impossible by the planarity and the fact that G has no separating triangle. If e_1, e_2 and another edge $e_3 \in E_4$ are on a separating 6-cycle of G, we get a similar contradiction. □

Fig. 7

The set E_4, however, needs not be compatible since the condition (3) of Definition 4.2 may fail. We will delete some edges from E_4 so that the resulting set $\overline{E}_4$ is compatible. Define:

$E_F = \{(x,u) \mid \exists (v,u) \in E_4$ and x is the neighbor of v not consecutive with $u\}$.

E_F is the set of the new edges that would have been obtained if E_4 were contracted from G. Let $F = (V_F, E_F)$ be the graph induced by E_F.

Lemma 4.9: F contains no cycles.

Proof: Suppose F has a cycle C_F consisting of vertices $\{u_1, \ldots, u_k\}$. Then G has a cycle C consisting of vertices $\{v_1, u_1, \ldots, v_k, u_k\}$, where $v_i \in I'_4$ and u_i is the selected contractable neighbor of v_i (Fig 8). For $i = 1 \ldots k$, let $C(v_i) = \{u_i, x_i, w_i, y_i\}$, where $w_i = u_{i-1}$. Thus u_i is a bad neighbor of v_i. Since u_1 is selected for v_1, there is a 2-path consisting of $\{x_1, t_1, y_1\}$ such that u_1 is in the interior of the quadrangle $\{v_1, x_1, t_1, y_1\}$. By the planarity, t_1 must be u_j for some $j > 1$. Consider the vertex v_j. Since u_j is selected for v_j, there is a 2-path consisting of $\{x_j, t_j, y_j\}$ such that u_j is in the interior of the quadrangle $\{v_j, x_j, t_j, y_j\}$. By the planarity, this is impossible. □

Fig 8

The *conflict graph* $\overline{F}$ is defined as follows. The node set of $\overline{F}$ is the edge set E_F of F. Two nodes e_1 and e_2 are adjacent in $\overline{F}$ if and only if e_1 and e_2 share a common vertex and the other two end vertices of e_1 and e_2 are adjacent in G. Since F is a forest, it is easy to show $\overline{F}$ is a planar graph. Let $\overline{K}$ be an independent set of $\overline{F}$. Define:

$\overline{E}_4 = \{(v, u) \in E_4 \mid$ the node in $\overline{F}$ corresponding to (x, u) is in $\overline{K}\}$.

Lemma 4.10: The set $\overline{E}_4$ is compatible.

Proof: $\overline{E}_4$ is obtained from E_4 by deleting some edges. Since E_4 satisfies the conditions (1), (2) and (4) of Definition 4.2, so does $\overline{E}_4$. We show $\overline{E}_4$ also satisfies the condition (3).

If two edges $e_1 = (v_1, u_1), e_2 = (v_2, u_2) \in E_4$ are on a separating 5-cycle of G, then v_1 and v_2 have a common neighbor x and (u_1, u_2) is an edge in G. Thus the nodes corresponding to e_1 and e_2 are adjacent in $\overline{F}$. Since $\overline{K}$ is independent, at most one of e_1 and e_2 is in $\overline{E}_4$. Hence no two edges in $\overline{E}_4$ are on a separating 5-cycle of G. □

4.4 The algorithm

We are now ready to present our algorithm.

Algorithm 4: Find a REL of a PTP graph G.

(1) If G has only one interior vertex, return the trivial REL and stop.

(2) Find a large sparse independent set $I = I_6 \cup I_5 \cup I_4$ as in Lemma 2.1.

(3) Examine the neighbors of each $v \in I_6$; construct E_6 and I''_6.

(4) Examine the neighbors of each $v \in I_5$; construct E_5 and I''_5.

(5) Examine the neighbors of each $v \in I'_4 = I_4 \cup I''_5 \cup I''_6$; construct E_4.

(6) Construct D and $\overline{D}$. Find a 7-coloring of $\overline{D}$. Let $\overline{J}$ be the largest color class. Construct $\overline{E}_5$.

(7) Construct F and $\overline{F}$. Find a 7-coloring of $\overline{F}$. Let $\overline{K}$ be the largest color class. Construct $\overline{E}_4$.

(8) Let E' be the largest set among E_6, $\overline{E}_5$ and $\overline{E}_4$.

(9) Construct the contracted graph $G' = G/E'$.

(10) Recurse on G' to find a REL of G'.

(11) Expand the REL of G' to be a REL of G.

End Algorithm 4.

We have shown each of $E_6, \overline{E}_5, \overline{E}_4$ is a set of compatible contractable edges of G and no two edges in E_j interfere with each other during expansion. So the algorithm is correct. We next analyze the complexity of Algorithm 4. In order to bound the recursion depth, we show $|E'| \geq c'n$ for some constant c'. First we note: $|I'_4| + |I'_5| + |I'_6| = |I| \geq cn$ and $|E_6| = |I'_6|$.

Consider the graph $\overline{D}$. Since $\overline{D}$ is a planar graph containing $|E_5|$ nodes and $\overline{J}$ is the largest color class of a 7-coloring of $\overline{D}$, we have $|\overline{E}_5| = |\overline{J}| \geq |E_5|/7 = |I'_5|/7$. Similarly, $|\overline{E}_4| = |\overline{K}| \geq |E_4|/7 = |I'_4|/7$. Hence $|E'| = \max\{\ |E_6|,\ |\overline{E}_5|,\ |\overline{E}_4|\ \} \geq \max\{|I'_6|, |I'_5|/7, |I'_4|/7\} \geq |I|/21 \geq c'n$ with $c' = c/21$.

Since each iteration removes a fixed fraction of vertices from G, the recursion depth of Algorithm 4 is $O(\log n)$. We show each iteration of Algorithm 4 can be implemented in $O(\log n)$ time with $O(n)$ processors. Step (1) takes $O(1)$ time with $O(1)$ processors. Step (2) can be done in $O(\log^* n)$ time with $O(n)$ processors [9]. Since the implementation of Steps (3), (4) and (5) is similar, we only discuss Step (4). First, for each $v \in V$, we construct an array $S(v)$ consisting of the neighbors of v sorted by vertex name (each vertex is represented by an integer between 1 and n). This pre-processing takes $O(\log n)$ time with $O(n)$ processors.

Step (4): For a given $v \in I_5$, we either select a good contractable neighbor of v, or determine this is impossible and put a suitable neighbor y of v into I''_5. The decision is based on two criteria: (i) whether a neighbor u of v is good; (ii) if u is good, whether it is connected by a 2-path to another neighbor w of v that is not consecutive with u. This can be done as follows: For each vertex x in $S(u)$, perform a binary search in $S(w)$ to see if x is also a neighbor of w. Since u has bounded degree (being a light vertex), this can be done in $O(\log n)$ time. Thus step (4) takes $O(\log n)$ time with $O(n)$ processors. Steps (3) and (5) can be implemented within the same time and processor bound.

Step (6): The graph D and $\overline{D}$ can be constructed in $O(\log n)$ time with $O(n)$ processors. A 7-coloring of the planar graph $\overline{D}$ can be constructed in $O(\log n)$ time with $O(n)$ processors [9, 11]. Hence this step totally takes $O(\log n)$ time with $O(n)$ processors. Step (7) can be implemented similarly.

Step (8) takes $O(\log n)$ time with $O(n)$ processors. As discussed in section 3.2, the contracted

graph G' can be constructed in $O(1)$ time with $O(n)$ processors. Step (10) is the recursive call. Step (11) takes $O(1)$ time with $O(n)$ processors.

Since the recursion depth is $O(\log n)$, Algorithm 2 takes $O(\log^2 n)$ time with $O(n)$ processors. This gives the following:

Theorem 4.11: A rectangular dual of a PTP graph G can be constructed in $O(\log^2 n)$ time with $O(n)$ processors.

Remark: Here we assume the topological ordering is used as the consistent numbering in Algorithm 3. If we use $level_1$ and $level_2$ as the consistent numbering of G_1^* and G_2^*, a more compact rectangular dual can be produced. However, the algorithm will need $M(n)$ processors.

Appendix: the proof of (1) in Lemma 2.1

Define and $n_i = |V_i|$ and $n_{[i,j]} = |V_{[i,j]}|$. Let $p_i = |U_i|$ $(i = 3, 4, 5)$. We need to show $n_3 + p_4 + p_5 + p_6 \geq n/24$. By Euler's formula we have: $3n_3 + 4n_4 + 5n_5 + 6n_6 + 7n_{[7,29]} + 30n_{[30,\infty]} \leq \sum_{v \in V_i} deg(v) = 2|E| \leq 6n$. This gives:

$$24n_{[30,\infty]} \leq 24n_{[30,\infty]} + n_{[7,29]} \leq 3n_3 + 2n_4 + n_5 \leq 3n \tag{1}$$

By counting the number of edges between $V_3 \cup (V_4 - U_4) \cup (V_5 - U_5) \cup (V_6 - U_6)$ and $V_{[30,\infty]}$, we get: $3n_3 + 3(n_4 - p_4) + 2(n_5 - p_5) + (n_6 - p_6) \leq \sum_{v \in V_{[30,\infty]}} deg(v)$, which gives: $4n_3 + 3n_4 + 2n_5 + n_6 - (n_3 + 3p_4 + 2p_5 + p_6) \leq \sum_{v \in V_{[30,\infty]}} deg(v)$. This in turn implies:
$4n_3 + 3n_4 + 2n_5 + n_6 + \sum_{v \in V_{[3,6]}} deg(v) + \sum_{v \in V_{[7,29]}} deg(v) - (n_3 + 3p_4 + 2p_5 + p_6) \leq \sum_{v \in V} deg(v)$
$= 2|E| \leq 6(n_3 + n_4 + n_5 + n_6 + n_{[7,29]} + n_{[30,\infty]})$. Simplifying this inequality, we get:

$n_3 + n_4 + n_5 + n_6 + n_{[7,29]} - (n_3 + 3p_4 + 2p_5 + p_6) \leq 6n_{[30,\infty]}$. Hence

$$n_3 + 3p_4 + 2p_5 + p_6 \geq n - 7n_{[30,\infty]} \tag{2}$$

From (1) and (2), we get: $n_3 + p_4 + p_5 + p_6 \geq \frac{1}{3}(n_3 + 3p_4 + 2p_5 + p_6) \geq \frac{1}{3}(n - \frac{21}{24}n) = \frac{1}{24}n$.

References

[1] K. Abrahamson, N. Dadoun, D. G. Kirkpatrick, and T. Przytycka, A Simple Tree Contraction Algorithm, J. Algorithms 10 (1989), pp. 287-302.

[2] J. Bhasker and S. Sahni, A linear Time algorithm to Check for the Existence of a Rectangular Dual of a Planar Triangulated Graph, Networks 17 (1987), pp. 307-317.

[3] J. Bhasker and S. Sahni, A linear algorithm to Find a Rectangular Dual of a Planar Triangulated Graph, Algorithmica 3 (1988), pp. 247-278.

[4] M. Chrobak and T. H. Payne, A Linear Time Algorithm for Drawing Planar Graphs on a Grid, Tech. Rep. UCR-CS-90-2, Department of Mathematics and Computer Science, University of California at Riverside, 1990.

[5] I. Fáry, On Straight Line Representation of Planar Graphs, Acta Sci. Math. Szeged 11 (1948), pp. 229-233.

[6] H. de Fraysseix, J. Pach and R. Pollack, Small Sets Supporting Fáry Embeddings of Planar Graphs, Proc. 20th ACM STOC (1988), pp. 426-433.

[7] M. Fürer and B. Raghavachari, On Computing Straight-line Embeddings of Planar Graphs Efficiently in Parallel, 1990, manuscript.

[8] H. Gazit and G. L. Miller, An Improved Parallel Algorithm That Computes the BFS Numbering of a Directed Graph, Info. Proc. Lett. 28 (1988), pp. 61-65.

[9] A. V. Goldberg, S. A. Plotkin and G. E. Shanon, Parallel Symmetry Breaking in Sparse Graphs, Proc. 19th ACM STOC (1987), pp. 315-324.

[10] X. He, On Finding the Rectangular Duals of Planar Triangulated Graphs (1990), Tech. Rep. 90-24, Department of Computer Science, State University of New York at Buffalo. Submitted for publication.

[11] X. He, Efficient Parallel and Sequential Algorithms for 4-Coloring Perfect Planar Graphs, Algorithmica 5 (1990), pp. 545-559.

[12] M-Y Kao and P. N. Klein, Towards Overcoming the Transitive-Closure Bottleneck: Efficient Parallel Algorithms for Planar Digraphs, in Proc. 22nd ACM STOC, 1990, pp. 181-192.

[13] K. Koźmiński and E. Kinnen, Rectangular Dual of Planar Graphs, Network 5 (1985), pp. 145-157.

[14] V. Ramachandran and J. Reif, An Optimal Parallel Algorithm for Graph Planarity, Proc. 30th IEEE FOCS (1989) pp. 282-287.

[15] W. Schnyder, Planar Graphs and Poset Dimension, Orders 5 (1989) pp. 323-343.

[16] W. Schnyder, Embedding Planar Graphs on the Grid, Proc. 1st ACM-SIAM Symposium on Discrete Alg. (1990) pp. 138-148.

[17] W. T. Tutte, A Theorem on Planar Graphs, Trans. Am. Math. Soc. 82 (1956), pp. 99-116.

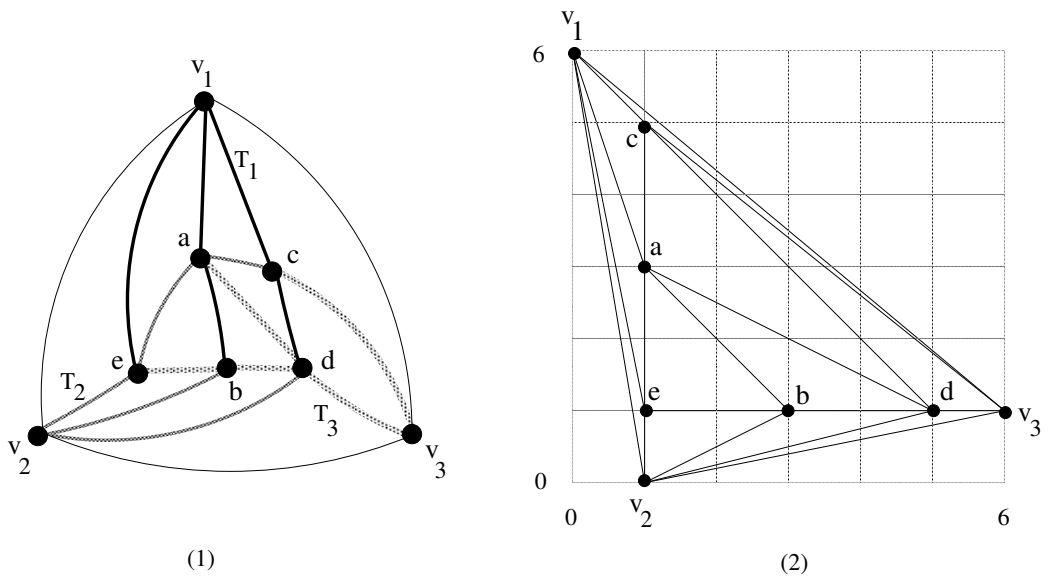

Figure 1. (1) A planar graph G, and (2) a Fary embedding of G.

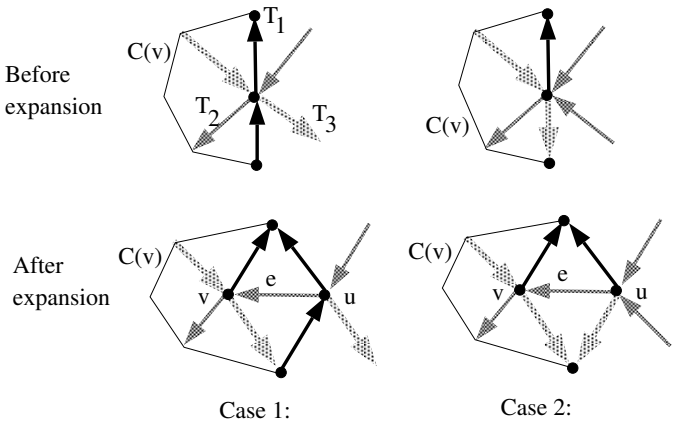

Figure 2. Examples of edge expansion.

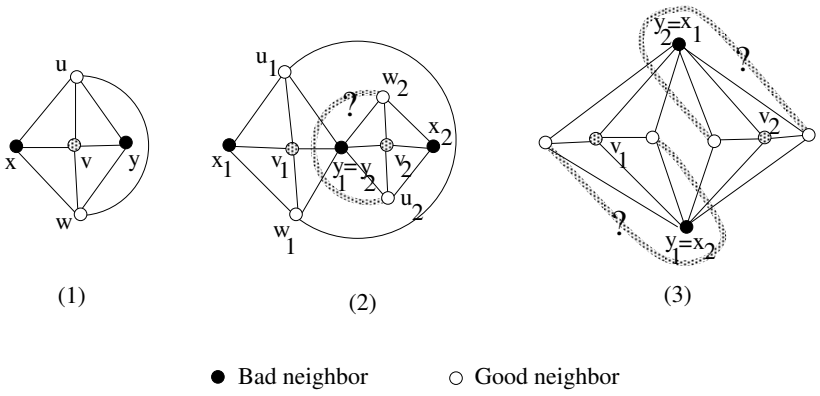

Figure 3. Select contractable neighbor for degree-4 vertices.

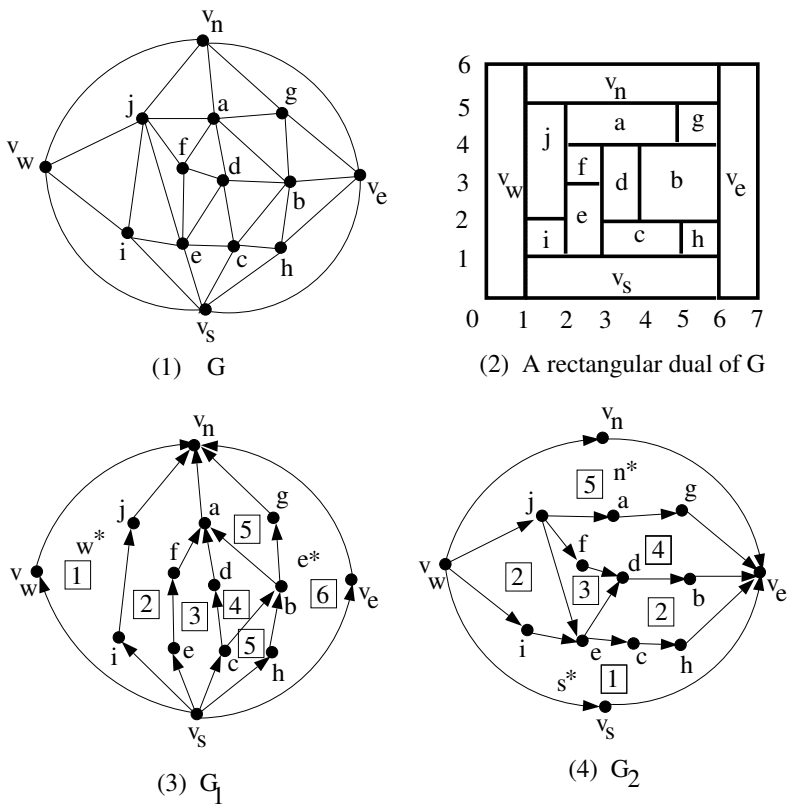

Figure 4. A graph G and its rectangular dual.

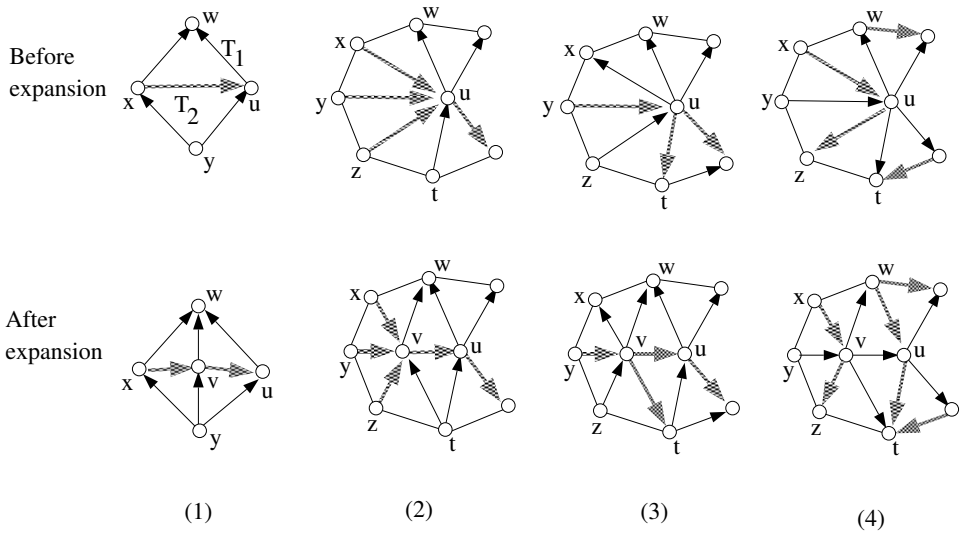

Figure 5. Edge expansion.

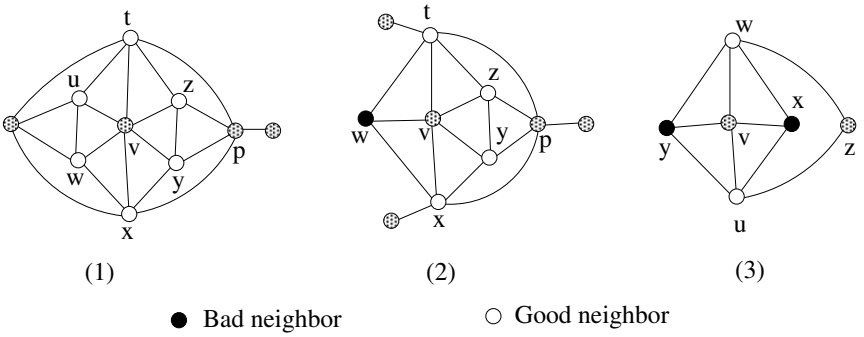

Figure 6. Select contractable neighbors.

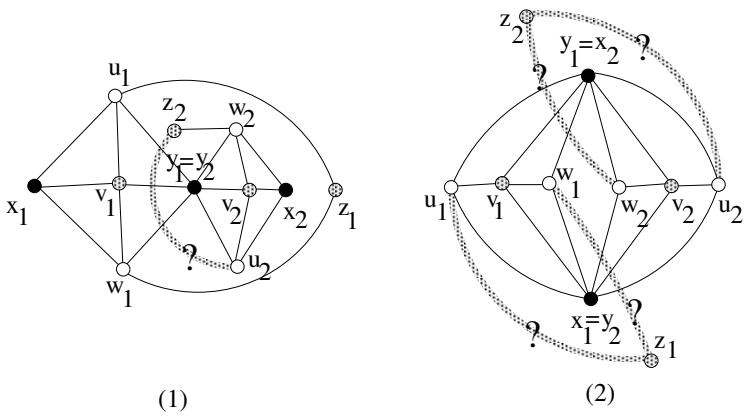

Figure 7. Proof of lemma 4.8.

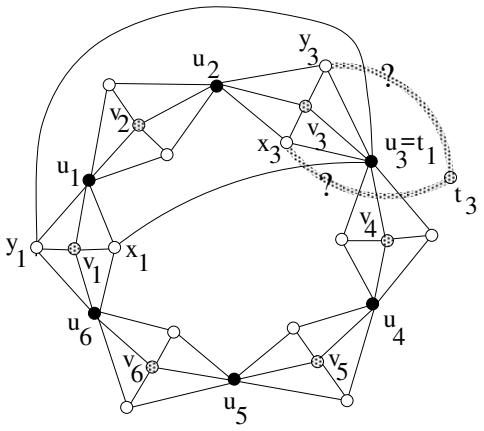

Figure 8. The proof of lemma 4.9.