1998 International Computer Symposium
Workshop on Algorithms
December 17-19, 1998, N.C.K.U., Tainan, Taiwan, R.O.C.

ON THE COMPLEXITY OF THE MAXIMUM BIPLANAR SUBGRAPH PROBLEM

Chang Wu Yu

Department of Information Management,
Minghsin Institute of Technology, Hsin-Chu, Taiwan, R. O. C.
Email: cwyu@mis.mhit.edu.tw

ABSTRACT

Let $G=(S, T, E)$ be a bipartite graph with vertex set $S \cup T$ and edge $E \subseteq S\times T$. A typical convention for drawing G is to put the vertices of S on a line and the vertices of T on a separate parallel line and then represent edges by placing straight line segments between the vertices that determine them. In this convention, a drawing is biplanar if edges do not cross, and a subgraph of G is biplanar if it has a biplanar drawing. The maximum biplanar subgraph problem is to find a biplanar subgraph with a maximum number of edges. In general, this maximum biplanar subgraph problem is NP-complete. In this paper, we show the maximum biplanar subgraph problem belongs to not only the class P, but also the class NC, when input graphs are restricted to doubly convex-bipartite graphs which is an important subclass of bipartite graphs. Moreover, our sequential algorithm is optimal.

Keywords: Maximum biplanar subgraph, graph drawing, doubly convex-bipartite graphs, P class, NC class, crossing number

SECTION 1. INTRODUCTION

Let $G=(S, T, E)$ be a bipartite graph with vertex set $S \cup T$ and edge $E \subseteq S\times T$. A typical convention for drawing G is to put the vertices of S on a line and the vertices of T on a separate parallel line and then represent edges by placing straight line segments between the vertices that determine them. In this convention, a drawing is *biplanar* if edges do not cross, and a subgraph of G is biplanar if it has a biplanar drawing (see Figure 1). The *maximum biplanar subgraph problem* is to find a biplanar subgraph with a maximum number of edges.

The maximum biplanar subgraph problem has been investigated in three application areas: automatic graph drawing of directed graph [8], global routing for row-based VLSI layout [11], and computational biology [18]. Traditional methods for drawing directed graphs are arranging nodes on levels, and apply a crucial crossing reduction step to reduce the number of edge crossings in between two successive levels [8]. But the crossing reduction step is NP-complete. Instead, Mutzel suggested the maximum biplanar subgraph problem could be used to increase the readability of diagram [13]. In standard cell technology for VLSI, modules are arranged in rows with wiring channels between each pair of rows. A planar subgraph of the net list represents a subset of the set of nets that can be routed in one layer [11]. In DNA mapping, small fragments of DNA have to be ordered according to the given overlap data and some additional information. If the overlap data is correct, the maps can be represented as biplanar graph. But, in practice, the overlap data may contain errors. Hence, Waterman and Griggs [18] suggested solving the maximum biplanar subgraph problem.

A biplanar graph can be recognized in linear time [17]. However, the maximum biplanar subgraph problem is NP-complete, even for the case when each vertex in S has degree three and each vertex in T has degree two [7]. Therefore, Eades and Whitesides suggested a heuristic based on the search for a longest path [7]. When the positions of all vertices are fixed, a maximum biplanar subgraph C of $G=(S, T, E)$ can be found in time $O(m\log r + n \log n)$, where $m = |E|$, $r=|C|$, $n=|S|+|T|$. In [14], Shahrokhi *et al.* computed the weighted maximum biplanar subgraph of forest by using a linear time algorithm. In [13], Mutzel gave an integer programming formulation for the weighted maximum biplanar subgraph problem. And by combing their cutting plane algorithm with a branch and bound algorithm, they found a solution that is close to the optimal one.

In this paper, we show the maximum biplanar subgraph problem for an important subclass of bipartite graphs, termed doubly convex-bipartite graphs, belongs to not only the class P, but also the class NC (Nick's class [9]). More specifically, we can solve this problem in $O(m)$ sequential time, and $O(\log n)$ time using $O(n^3/\log n)$ processors on the CRCW PRAM or $O(\log^2 n)$ time using $O(n^3/\log^2 n)$ processors on the CREW PRAM, where n, m is the number of vertices and edges of input graph, respectively. Note that the double convex-bipartite graph is not a subclass of forest and our sequential algorithm is optimal.

This reminder of this paper is organized as follows. In the next section, we show some basic definitions and properties. In section 3, the main theorem of this paper are shown, In section 4, our sequential and parallel algorithms for the maximum biplanar subgraph problem are described and analyzed. Finally, some concluding remarks are given in Section 5.

SECTION 2. DEFINITIONS AND PROPERTIES

There are some properties characterize biplanar graphs. A *caterpillar* is a connected graph that has a path called the *backbone* b such that all vertices of degree larger than one lie on b. The edges of a caterpillar that are not on the backbone are the *legs* of the caterpillar. See Figure 2 for example.

Lemma 1. (Eades *et al.* [6]) A bipartite graph is biplanar if and only if it is a collection of disjoint caterpillars.

We call the graph shown in Figure 3 a *double claw*.

Lemma 2. (Eades *et al.* [6]). A bipartite graph is biplanar if and only if it contains no cycle and no double claw.

Lemma 3. (Tommi *et al.* [17]). A bipartite graph is biplanar if and only if the graph G^* that is the remainder of G after deleting all vertices of degree one, is acyclic and contains no vertices of degree at least three.

Now we define the doubly convex-bipartite graph. Let $G=(S, T, E)$ be a bipartite graph with vertex set $S\cup T$ and edge $E\subseteq S\times T$. Also, let $N(v)$ denote the set of vertices which are adjacent to v in G. An ordering of $S(T)$ has the *adjacency property* if for each vertex $v\in T(S)$, $N(v)$ contains consecutive vertices in this ordering. The graph $G=(S, T, E)$ is called a *doubly convex-bipartite graph* if there are orderings of S and T having the adjacency property [12]. See Figure 4, where a doubly convex-bipartite graph is shown. Glover [10] showed a practically important application of doubly convex-bipartite graphs in industry. Lipski and Preparata [12] solved the recognition problem and the maximum matching problem on doubly convex-bipartite graphs in linear time. In [19], Yu and Chen proposed a parallel algorithm to recognize the doubly convex-bipartite graph in $O(\log n)$ time using $O(n^3/\log n)$ processors on the CRCW PRAM, or $O(\log^2 n)$ time using $O(n^3/\log^2 n)$ processors on the CREW PRAM.

The combination of an ordering of S and an ordering of T is called a *strong ordering* if any two edges (s_w, t_x), $(s_y, t_z)\in E$ imply $(s_w, t_z)\in E$ and $(s_y, t_x)\in E$, where $s_w, s_y\in S$, $t_x, t_z\in T$, s_w precedes s_y in the ordering of S, and t_z precedes t_x in the ordering of T. A graph is a *bipartite-permutation graph* $G=(S, T, E)$ if and only if there is a strong ordering of S and T [16]. Other equivalent definitions of bipartite-permutation graphs can be found in [16].

Suppose that U and V are nonempty subsets of S and T, respectively. The subgraph of G whose vertex set is $U\cup V$ and whose edge set contains those edges of G that have both ends in $U\cup V$ is called the subgraph of G *induced* by U and V, and is denoted by $G_{U,V}$.

Notations and definitions not stated explicitly can be found in [1].

SECTION 3. MAIN THEOREM

In this section, we want to describe some properties of doubly convex-bipartite graphs that will be used to construct a maximum biplanar subgraph. Suppose that $G=(S, T, E)$ is a connected doubly convex-bipartite graph, and $s_1, s_2, \ldots, s_{|S|}$ and $t_1, t_2, \ldots, t_{|T|}$ are orderings of S and T, respectively, having the adjacency property. We call a vertex $v\in S\cup T$ is *maximal* if there does not exist $u\in S\cup T$ such that $N(v)$ is a proper subset of $N(u)$. For example, those maximal vertices of G (in Figure 4) are s_4, s_6, s_7, t_2, t_3, t_4, t_5, and t_6.

Let S_M and T_M are sets of those maximal vertices in S and T respectively. And let s_l and s_r are the first and the last maximal vertices (with respect to the ordering of S, that is $s_1, s_2, \ldots, s_{|S|}$) in S_M. Similarly, we can define t_L and t_R are the first and the last maximal vertices(with respect to the ordering of T, that is $t_1, t_2, \ldots, t_{|T|}$) in T_M.

Lemma 4 (Lemma 8 in [19]): Suppose that $G=(S, T, E)$ is a connected doubly convex-bipartite graph. Then, we have $G_{S_M, T}$ is a connected bipartite-permutation graph, and its corresponding ordering is a strong ordering. (Note that, Yu *et al.* use the symbol A to stand for S_M in [19].)

Lemma 5: Suppose that $G=(S, T, E)$ is a connected doubly convex-bipartite graph. Then, we have $(s_l, t_L)\in E$ and $(s_r, t_R)\in E$.

Proof: Since $G=(S, T, E)$ is a connected doubly convex-bipartite graph. By Lemma 4, we have $G_{S_M, T}$ is a connected bipartite-permutation graph. Because the induced subgraph of the bipartite-permutation graph is also a bipartite-permutation graph, we have G_{S_M,T_M} is a bipartite-permutation graph. The connectivity of G_{S_M,T_M} is discussed as follows. Suppose to the contrary, G_{S_M,T_M} is disconnected. Then S_M can be partitioned into $S_{M1}, S_{M2}, \ldots, S_{Mx}$, where $x\geq 2$, and each of which is contained in a distinct component of G_{S_M,T_M}. Then there must exist edges (s_j, t_i) and (s_k, t_i) in the edge set of $G_{S_M, T}$, where $s_j\in S_{Mw}$, $s_k\in S_{My}$, $j\neq k$, $y\neq w$, and $t_i\in T-T_M$. But there must exist $t_z\in T_M$ such that $N(t_i)\subseteq N(t_z)$. So (s_j, t_z) and (s_k, t_z) are in the edge set of G_{S_M,T_M}. Then we have S_{Mw} and S_{My} are contained in the same component of G_{S_M,T_M}. A contradiction occurs. So G_{S_M,T_M} is connected. Based on above discussions, we have G_{S_M,T_M} is a connected bipartite-permutation graph.

Let E_M is the edge set of G_{S_M,T_M}. Suppose $(s_l, t_L)\notin E$. Then $(s_l, t_L)\notin E_M$. By the definition of s_l and t_L, and the fact that G_{S_M,T_M} is connected. We can assume there is an edge $(s_l, t_j)\in E_M$ and $j > L$, and $(s_i, t_L)\in E_M$ and $i > l$ (see Figure 5). We have (s_l, t_j) crosses (s_i, t_L). Because G_{S_M,T_M} is a bipartite-permutation graph, we have $(s_l, t_L)\in E_M$ and then $(s_l, t_L)\in E$ by the definition of the strong ordering,. This is a contradiction. So, we have $(s_l, t_L)\in E$.

Similarly, we can conclude $(s_r, t_R)\in E$. □

Suppose $s_1, s_2, \ldots, s_l, \ldots, s_r, \ldots, s_{|S|}$ and $t_1, t_2, \ldots, t_L, \ldots, t_R, \ldots, t_{|T|}$ are the orderings of S and T which have the adjacency property. Let $G^* =(S^*, T^*, E^*)$, where $S^*=\{ s_l, s_{l+1}, \ldots, s_r\}$, $T^*=\{ t_L , t_{L+1} \ldots, t_R\}$ and $E^*=E\cap S^*\times T^*$. That is, G^* is the induced subgraph of G by the vertex set $S^* \cup T^*$(see Figure 6). Where l=4, r=7, L=2, and R=6.

Lemma 6: Suppose that G=(S, T, E) is a connected doubly convex-bipartite graph. Then, G^* is a connected bipartite-permutation graph.

Proof: To show G^* is a bipartite-permutation graph, it is sufficient to prove that $G_{S^*, T}$ is a bipartite-permutation graph. The connectivity of G^* is assured by the facts that G_{S_M, T_M} is an induced subgraph of G^* and G_{S_M, T_M} is connected by the discussions in the first paragraph of the proof of Lemma 5.

To show $G_{S^*, T}$ is a bipartite-permutation graph, we transform a convex bipartite graph to a rectilinear polygon. We only need to place a square at the position (t_i, s_j) for each edge (s_j, t_i) $\in E$. The polygon can be partitioned into three parts: bottom region, middle region, and the top region. Moreover, the middle region consists of three parts: upper part, center part, and lower part (See Figure 7 for example). It is easy to see that the corresponding part of $G_{S^*, T}$ is the center part of the middle region [19]. And the corresponding subgraph(that is, the graph $G_{A\cup C, T}$ in [19]) of the middle region is shown to be a connected bipartite-permutation graph by Lemma 9 in [19]. So the corresponding subgraph of the center part of the middle region, that is $G_{S^*, T}$, is also a bipartite-permutation graph.

Finally, we conclude that G^* is a connected bipartite-permutation graph. □

Here we say a path is *noncrossing* if it is a path without any crossing when embedding in the orderings of S and T, that is $s_1, s_2, \ldots, s_{|S|}$ and $t_1, t_2, \ldots, t_{|T|}$, having the adjacency property.

Lemma 7(Claim in [16]): Let G=(S, T, E) be a bipartite-permutation graph which contain a Hamiltonian path beginning at vertex s in S. Then $s_1, t_1, s_2, t_2, \ldots, s_k, t_k$ (followed by s_{k+1} if it exists) is also a Hamiltonian path in G, where the orderings of S and T, that is $s_1, s_2, \ldots, s_k$ (followed by s_{k+1} if it exists); $t_1, t_2, \ldots, t_k$, is a strong ordering of G.

Note that this Hamiltonian path is noncrossing.

Lemma 8: Suppose that G=(S, T, E) is a connected doubly convex-bipartite graph. Then, there is a noncrossing path in G^* which connects (s_l, t_L) and (s_r, t_R).

Proof: The existences of (s_l, t_L) and (s_r, t_R) are assured by Lemma 5. Since G^* is connected (by Lemma 6), there is a path P=(S_p, T_p, E_p) connected (s_l, t_L) and (s_r, t_R). Suppose P has some crossings in the drawing of G^*. We try to augment P by the following method. Then we may find another path which is noncrossing in G^* and connects (s_l, t_L) and (s_r, t_R).

For any two edges (s_w, t_x) and (s_y, t_z) in E_p which form a crossing, we add (s_w, t_z) and (s_y, t_x) to E_p. Note that (s_w, t_z) and (s_y, t_x) are in E^* by Lemma 6 and the strong ordering property of bipartite-permutation graphs. Then we check if there exist two edges in the new E_p which form new crossings. If so, add more edges by the similar approach. Eventually, no new crossing occurs. Let the resulting graph is G_c=(S_c, T_c, E_c). Then G_c is a subgarph of G^* and G_c is a bipartite-permutation graph by the strong ordering property. Note that G_c and P share the same vertex set. Because P is a Hamiltonian path in G_c, G_c is hamiltonian. And by Lemma 7, We can obtain a noncrossing path in G^* which connects (s_l, t_L) and (s_r, t_R). □

Next, we want to construct a subgraph C of G, and finally prove that it is a maximum biplanar subgraph of G.

Suppose P is the noncrossing path which connects (s_l, t_L) and (s_r, t_R). Without loss of generality, let P=(($s_l = s_{o(1)}$, $t_L = t_{q(1)}$), ($s_{o(2)}$, $t_{q(1)}$), ($s_{o(2)}$, $t_{q(2)}$), ..., ($s_{o(k-1)}$, $t_{q(k-1)}$), ($s_{o(k)}$, $t_{q(k-1)}$), ($s_{o(k)}=s_r$, $t_{q(k)}=t_R$)). You may view P as the backbone of the caterpillar. Then, we want to append legs to the caterpillar by the following edge sets. Note that all the following edge sets belongs to the edge set E.

Let EP is the edge set of P. And let
$ES_1=\{(s_{o(1)}= s_l, t_j) \mid \text{where } 1\le j\le L-1\}$,
$ES_i=\{(s_{o(i)}, t_j) \mid \text{where } q(i-1)<j<q(i)\}$, for all $1<i\le k-1$,
$ES_k=\{(s_{o(k)}= s_r, t_j) \mid \text{where } q(k-1)<j<q(k)=R \text{ or } q(k) =R <j\le |T|\}$.
Similarly, we define
$ET_1=\{(s_i, t_L=t_{q(1)}) \mid \text{where } 1\le i\le l-1 \text{ or } l= o(1) <i<o(2)\}$,
$ET_j=\{(s_i, t_{q(j)}) \mid \text{where } o(i)<i<o(i+1)\}$, for all $1<j\le k-1$,
$ET_k=\{(s_i, t_R=t_{q(k)}) \mid \text{where } r<i\le |S|\}$.
Finally, let $C=EP\cup(\cup ES_i$, for $1\le i\le k)\cup(\cup ET_j$, for $1\le j\le k)$. See Figure 8 for example. The edge set EP of the noncrossing path P is {(s_4, t_2), (s_5, t_2), (s_5, t_4), (s_7, t_4), (s_7, t_6)}. And ES_1={(s_4, t_1)}, ES_2={(s_5, t_3)}, ES_3={(s_7, t_5), (s_7, t_7), (s_7, t_8)}, ET_1={(s_1, t_2), (s_2, t_2), (s_3, t_2)}, ET_2={(s_6, t_4)}, ET_3={(s_8, t_6), (s_9, t_6), (s_{10}, t_6)}.

Theorem 1: Suppose that G=(S, T, E) is a connected doubly convex-bipartite graph. Then, C is a maximum biplanar subgraph of G.

Proof: At first, we prove C is the biplanar subgraph of G. Then, we show its edge size is maximum. The existence of P is assured by Lemma 8. Since $EP\subseteq E$, $ES_i\subseteq E$ for $1\le i\le k$, and $ET_j\subseteq E$ for $1\le j\le k$, we have C is a subgraph of G. And the remainder of C after deleting all vertices of degree one, that is all ES_i and ET_j, is a path. A path is acyclic and contains no vertices of degree at least three. So by Lemma 3, C is a biplanar subgraph.

Since a biplanar subgraph is a tree. And the size of its edge set is the size of vertex set minuses one. So the size of vertex set of any maximum biplanar subgraph of G are not greater than the size of $S\cup T$. The vertex set of C is exactly $S\cup T$, so C is a maximum biplanar subgraph of G.

□

SECTION 4. ALGORITHM AND COMPLEXITIES

According the discussions of previous section, the following algorithm can solve the maximum biplanar subgraph problem on the connected doubly convex-bipartite graph.

Input: A connected doubly convex-bipartite graph $G=(S, T, E)$.

Output: Find the maximum biplanar subgraph C of G and its biplanar drawing.

Step 1. Determine the orderings of S and T that have the adjacency property.

Step 2. Find these two edges (s_l, t_L) and (s_r, t_R).

Step 3. Find a noncrossing path P which connects (s_l, t_L) and (s_r, t_R) in G^*.

Step 4. Construct C and its biplanar drawing.

By Theorem 1 C is a maximum biplanar subgraph of G. Complexity of the algorithm is analyzed as follows. Let n and m denotes the numbers of vertices and edges, respectively, in G. Step 1 can be completed in $O(m)$ sequential time by Booth and Lueker's work [2]. If Yu and Chen's algorithm is applied, Step 1 can be completed in $O(\log n)$ time using $O(n^3/\log n)$ processors on the CRCW PRAM, or $O(\log^2 n)$ time using $O(n^3/\log^2 n)$ processors on the CREW PRAM.

Step 2 can be solved by the following substeps. Let us define $Left(s_i)=min\{j \mid t_j \in N(s_i)\}$ and $Right(s_i) = max\{j \mid t_j \in N(s_i)\}$, where $s_i \in S$.

Step 2.1 Compute $Left(s_i)$ and $Right(s_i)$, for all s_i in S.

Step 2.2 Find $A=\{s_j \mid Left(s_j)$ is the smallest in $S\}$.

Step 2.3 Find $B=\{s_i \mid Right(s_i)$ is the largest in $A\}$.

Step 2.4 Find s_l, where $l=min\{i \mid S_i \in B\}$.

Similar steps can be used to compute s_r, t_L, and t_R. And all this four substeps can be implemented in $O(m)$ sequential time, and $O(\log n)$ time using $O(n^2)$ processors on the CREW PRAM.

Step 3 can be solved by the methods described in Lemma 8.

Step 3.1 find an arbitrary path in G^* which connects (s_l, t_L) and (s_r, t_R).

Step 3.2 augments this path into a bipartite-permutation subgraph by the following method. Let E_p be the edge set of this path. For any two edges (s_w, t_x) and (s_y, t_z) in E_p which form a crossing, we add (s_w, t_z) and (s_y, t_x) to E_p. Then we check if there exist two edges in the new E_p which form new crossings. If so, add more edges by the similar approach. Repeat this step until no new crossing occurs.

Step 3.3 construct a Hamiltonian path in this resulting graph.

Step 3.1 can be executed parallelly by applying Chen's all pairs shortest paths algorithm [4] which takes $O(\log n)$ time using $O(n^2/\log n)$ processors on the CREW PRAM. Because G^* is a bipartite-permutation graph by Lemma 6, Step 3.2 can be computed by finding a subgraph induced by the vertex set of the path, which takes $O(\log n)$ time using $O(n^2)$ processors on the CREW PRAM. Step 3.3 can be implemented easily according to Lemma 7 in $O(\log n)$ time using $O(n/\log n)$ processors on the EREW PRAM. So Step 3 can be implemented in $O(\log n)$ time using $O(n^2)$ processor on the CREW PRAM. Similarly, Step 3 can be implemented in $O(m)$ sequential time.

Step 4 can be implemented easily in the same complexity as Step 3.

Totally, this algorithm need $O(m)$ sequential time, and $O(\log n)$ time using $O(n^3/\log n)$ processors on the CRCW PRAM or $O(\log^2 n)$ time using $O(n^3/\log^2 n)$ processors on the CREW PRAM. Note that our sequential algorithm is optimal.

Theorem 2: If G is a connected doubly convex-bipartite graph, the maximum biplanar graph problem belongs to not only the class P, but also the class NC. More specifically, we can solve the problem in $O(m)$ sequential time, and $O(\log n)$ time using $O(n^3/\log n)$ processors on the CRCW PRAM or $O(\log^2 n)$ time using $O(n^3/\log^2 n)$ processors on the CREW PRAM, where n is the size of vertex set and m is the size of edge set.

SECTION 5. CONCLUSION AND FURTHER RESEARCHES

In Section 3, we have assumed that the input graph is connected. In fact, the restriction to be connected graphs can be removed. If the input graph is not connected, we simply execute the proposed algorithm for each of its component. If we apply Shiloach and Vishkin's connected component algorithm [15], the time and processor complexities required remain the same for a disconnected input graph.

The containment relationships for some subclasses of bipartite graphs are known as follow [3]: bipartite-permutation graphs $\subset$ doubly convex-bipartite graphs $\subset$ convex-bipartite graphs $\subset$ chordal bipartite graphs $\subset$ perfect elimination bipartite graphs $\subset$ bipartite graphs. The maximum biplanar subgraph problem has been proved to be NP-complete for the bipartite graph. In this paper we further show that it belongs to the class P and class NC for the doubly convex-bipartite graph. The immediate open problem behind this paper is to decide whether the maximum biplanar subgraph problem is solvable in polynomial time for convex bipartite graphs.

REFERENCES

1. J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*, Macmillan, London, 1976.
2. K. S. Booth and G. S. Lueker, "Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithm," *J. Comput.*

System Sci, vol. 13, pp. 335-379, 1976.

3. A. Brandstadt, J. Spinrad and L. Stewart, "Bipartite permutation graphs are bipartite tolerance graphs," *Congressus Numerantium*, vol. 58, pp. 165-174, 1987.
4. L. Chen, "Solving the shortest-paths problem on bipartite permutation graphs efficiently," *Information Processing Letters*, vol. 55, no. 5, pp. 259-264, 1995.
5. P. Eades and D. Kelly, "Heuristic for reducing crossings in a 2-layered networks," *Ars Combin.* Vol. 21a, pp. 89-98, 1986.
6. P. Eades, B. D. McKay, and N. C. Wormald, "On an edge crossing problem," *Proc. 9th Australian Computer Science Conference*, Australian National University, 1986, pp. 327-334.
7. P. Eades and S. Whitesides, "Drawing graphs in two layers," *Theoretical Computer Science*, vol. 131, pp. 361-374, 1994.
8. P. Eades and N. C. Wormald, "Edge crossings in drawings of bipartite graphs," *Algorithmica*, vol. 11, pp. 379-403, 1994.
9. A. Gibbons and W. Rytter, *Efficient Parallel Algorithms*, Cambridge University Press, Cambridge, 1988.
10. F. Glover, "Maximum matching in a convex bipartite graph," *Naval Res. Logist. Quart.*, vol. 14, pp. 313-316, 1967.
11. T. Lengauer, *Combinatorial algorithms for integrated circuit layout*, John Wiley & Sons, Chichester, UK, 1990.
12. J. W. Lipski and F. P. Preparata, "Efficient algorithms for finding maximum matchings in convex bipartite graphs and related problem," *Acta Inform.*, vol. 15, pp. 329-346, 1982.
13. P. Mutzel, "An alternative method to crossing minimization on hierarchical graphs," *Symposium on Graph Drawing, Lecture Notes in Computer Sciences*, vol. 1190, 1996, pp. 318-333.
14. F. Shahrokhi, O. Sykora, L. A. Szekely, and I. Vrt'o, "On bipartite crossings, largest biplanar subgraphs, and the linear arrangement problem," *Symposium on Graph Drawing, Lecture Notes in Computer Sciences*, vol. 1190, 1996, pp. 55-58.
15. Y. Shiloach and U. Vishkin, "An $O(\log n)$ parallel connectivity algorithm," *Journal of Algorithms*, vol. 3, pp. 57-67, 1982.
16. J. Spinrad, A. Brandstadt, and L. Stewart, "Bipartite permutation graphs," *Discrete Applied Mathematics*, vol. 18, pp. 279-292, 1987.
17. N. Tomii, Y. Kambayashi, and Y. Shuzo, "On planarization algorithms of 2-level graphs," *Papers of tech. group on electronic computers, IECEJ*, EC77-38, pp. 1-12, 1977.
18. M. S. Waterman and J. R. Griggs, "Interval graphs and maps of DNA," *Bull. Math. Biology*, vol. 48, no. 2, pp. 189-195, 1986.
19. C. W. Yu and G. H. Chen, "Efficient parallel algorithms for doubly convex-bipartite graphs," *Theoretical Computer Science*, vol. 147, pp. 249-265, 1995.

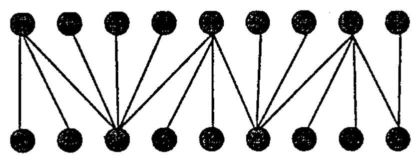

Figure 1. A biplanar graph.

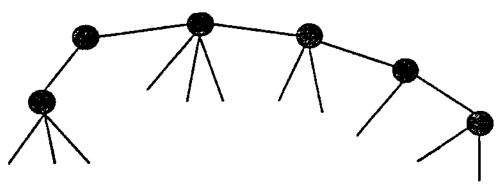

Figure 2. A caterpillar.

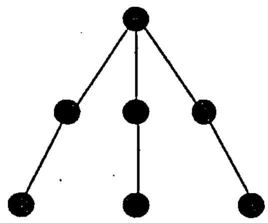

Figure 3. A double claw.

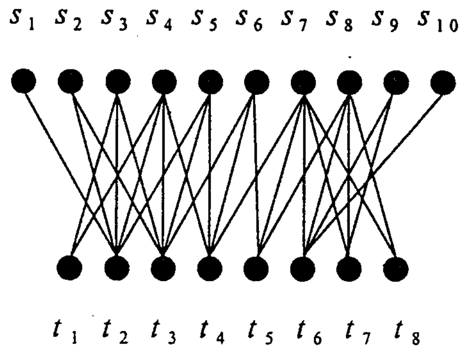

Figure 4. A doubly convex bipartite graph.

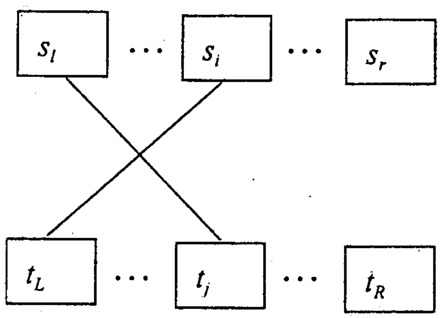

Figure 5.

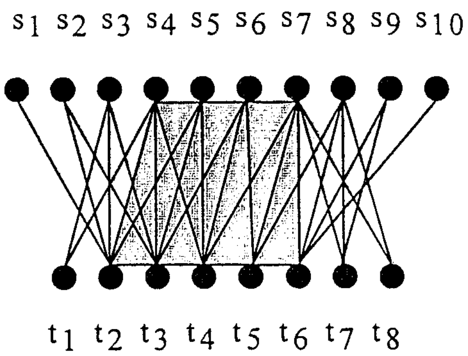

Figure 6. The corresponding G^* of Figure 4.

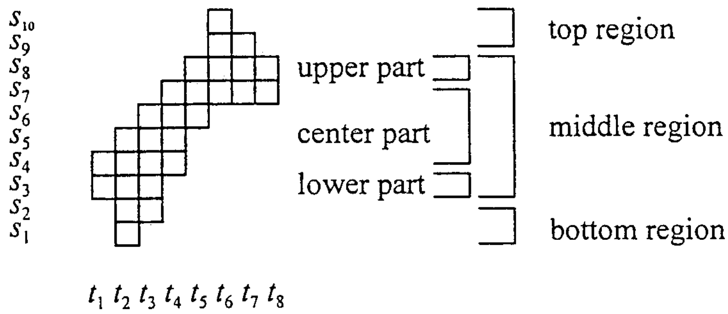

Figure 7. The rectilinear polygon constructed from the graph of Figure 4.

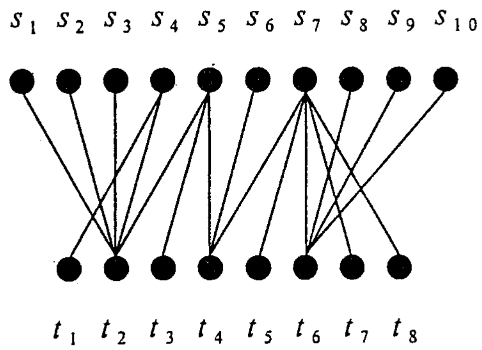

Figure 8. The corresponding C of G in Figure 4.