

**ALGORITHMS FOR EMBEDDING
GRAPHS IN BOOKS**

by

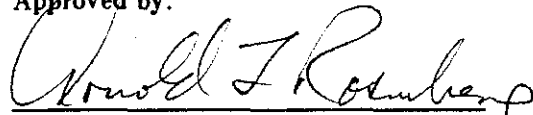
Lenwood Scott Heath

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.


Chapel Hill

1985


Approved by:



Advisor: Arnold Rosenberg



Reader: David Plaisted



Reader: Donald Stanat

© 1985
Lenwood Scott Heath
ALL RIGHTS RESERVED

LENWOOD SCOTT HEATH. Algorithms for Embedding Graphs in Books (Under the direction of ARNOLD L. ROSENBERG, Duke University.)

ABSTRACT

We investigate the problem of embedding graphs in books. A *book* is some number of half-planes (the *pages* of the book), which share a common line as boundary (the *spine* of the book). A *book embedding* of a graph embeds the vertices on the spine in some order and embeds each edge in some page so that in each page no two edges intersect. The *pagenumber* of a graph is the number of pages in a minimum-page embedding of the graph. The *pagewidth* of a book embedding is the maximum cutwidth of the embedding in any page. A practical application is in the realization of a fault-tolerant array of VLSI processors.

Our results are efficient algorithms for embedding certain classes of planar graphs in books of small pagenumber or small pagewidth.

The first result is a linear time algorithm that embeds any planar graph in a book of seven pages. This establishes the smallest upper bound known for the pagenumber of the class of planar graphs. The algorithm uses three main ideas. The first is to *level* the planar graph. The second is to *extend a cycle* at one level to the next level by doing *micro-surgery*. The third is to *nest* the embedding of successive levels to obtain finite pagenumber.

The second result is a linear time algorithm that embeds any trivalent planar graph in a book of two pages. The algorithm edge-augments the graph to make it hamiltonian while keeping it planar.

The third result is an $O(n \log n)$ time algorithm for embedding any outerplanar graph with small pagewidth. Our algorithm embeds any d -valent outerplanar graph in a two-page book with $O(d \log n)$ pagewidth. This result is optimal in pagewidth to within a constant factor. The significance for VLSI design is that any outerplanar graph can be implemented in small area in a fault-tolerant fashion.

ACKNOWLEDGEMENTS

I dedicate this dissertation to my wife Sheila, who is my reason for persevering. In the past four years, she has been my source of love, joy, support, inspiration, and meaning.

I express endless gratitude to Arnold Rosenberg for the hours he has devoted to my education and for the belief he expressed in my abilities. Without him, this work would not have been possible. He is also responsible for pointing me in the direction which I now follow.

The time and patience of the other members of my committee, Dean Brock, Tom Brylawski, Kye Hedlund, David Plaisted, and Don Stanat, are much appreciated. Don Stanat was my faithful guide through the perils of my graduate career. I especially thank Tom Brylawski for giving me the combinatorist's view of things.

My mother has always had more confidence in me than I have. Her unfailing love and belief traveled with me along the road to this accomplishment.

I thank the National Science Foundation (grant MCS-83-01213) and the Semiconductor Research Corporation (grant 41258) for support of this research.

TABLE OF CONTENTS

1 THE PROBLEM AND ITS MOTIVATIONS	1
1.1 The Problem	1
1.2 Motivations	3
1.2.1 Multilayer VLSI Layout	3
1.2.2 Design of Fault-Tolerant Processor Arrays	5
1.2.3 Sorting with Parallel Stacks	6
1.3 Structure of the Dissertation	7
2 PREVIOUS RESULTS AND TOOLS	8
2.1 Previous Results	8
2.1.1 Circular Embedding	9
2.1.2 One-Page Graphs	11
2.1.3 Two-Page Graphs	11
2.1.4 Planar Graphs	12
2.2 Tools	12
3 EMBEDDING PLANAR GRAPHS IN SEVEN PAGES	14
3.1 Overview of the Algorithm	14
3.2 Previous Approaches	19
3.2.1 The Bernhart-Kainen Conjecture	19
3.2.2 Bifurcators	25

3.2.3 Separating Triangles	26
3.3 Elements of the Seven-Page Algorithm	27
3.3.1 Levels	27
3.3.2 D-Cycles	35
3.3.3 Nesting	39
3.4 Levels Without Cycles	44
3.5 The Algorithm	46
3.5.1 The Statement	49
3.5.2 Why Seven?	54
3.5.3 Further Analysis	55
3.6 Conclusions	56
4 EMBEDDING TRIVALENT PLANAR GRAPHS IN TWO PAGES	58
4.1 Overview of the Algorithm	58
4.2 Structure of Biconnected Planar Graphs	61
4.3 The Main Theorem	68
4.4 The Algorithm	73
4.5 Oriented Face Traversal	77
4.6 Conclusions	82
5 EMBEDDING OUTERPLANAR GRAPHS IN SMALL BOOKS	84
5.1 Tradeoffs	84
5.2 Overview of the Algorithm	88
5.2.1 String Construction	92
5.2.2 Ladder Construction	95
5.3 The Algorithm	101
5.4 Correctness	105

5.5 Performance	110
5.6 Conclusion	111
6 CONCLUSIONS	112
REFERENCES	116
GLOSSARY	119

FIGURES

Figure 1.1. Grid Graph G	2
Figure 1.2. Two-Page Embedding of G	3
Figure 3.1. Sample Planar Graph G	15
Figure 3.2. Embedding of Level 0	16
Figure 3.3. Extended Cycle	17
Figure 3.4. Embedding of Levels 0 And 1	18
Figure 3.5. Embedding of Levels 0, 1, And 2	18
Figure 3.6. $ST(K_3)$	20
Figure 3.7. Three-Page Embedding of $ST(K_3)$	21
Figure 3.8. Inductive Hypothesis for Theorem 3.1	22
Figure 3.9. Inductive Step—Add r , s And t	23
Figure 3.10. Interior of $ST^2(K_3)$	24
Figure 3.11. Book Embedding of $ST^2(K_3)$	25
Figure 3.12. I-graph with Two Levels	29
Figure 3.13. I-Graph with Three Levels	31
Figure 3.14. Empty X_0	32
Figure 3.15. Example of Connected G_1	33
Figure 3.16. The BC-Graph of G_1	33
Figure 3.17 Visiting the Vertices of P_v	37

Figure 3.18. Supercycle Fragments (k)-Cycles	38
Figure 3.19. Micro-Surgery on a (k)-Cycle	40
Figure 3.20. Vertex z Shared by Two (k)-Cycles	41
Figure 3.21. The BC-Tree	42
Figure 3.22. Contraction of Cycle A to v_A	42
Figure 3.23. Contraction of Cycle A' Except z to $v_{A'}$	43
Figure 3.24. Contradiction for Theorem 3.14	45
Figure 3.25. Retriangulated Graph	47
Figure 3.26. Hamiltonian Cycle	48
Figure 3.27. K Before $G_{k+1} K$ Is Embedded	51
Figure 3.28. The Embedding of S	52
Figure 3.29. A Cylinder of Three Triangles	55
Figure 4.1. Biconnected Trivalent Planar Graph	59
Figure 4.2. Creation of Face F	60
Figure 4.3. Superhamiltonian Cycle H'	60
Figure 4.4. Class I Face	62
Figure 4.5. Class II Face	62
Figure 4.6. Class III Face	63
Figure 4.7. Dual Graph G^D	64
Figure 4.8. Curve Through a Class III Face	66
Figure 4.9. $G^L = G - F$	68
Figure 4.10. Proof of Lemma 4.7	69
Figure 4.11. Proof of Theorem 4.8	70
Figure 4.12. The Superhamiltonian Cycle H'	71
Figure 4.13. The Supercycle H''	72

Figure 4.14. The Superhamiltonian Cycle H	73
Figure 4.15. Oriented Face Traversal	78
Figure 4.16. Proof of Lemma 4.15	80
Figure 5.1. The 7-Ladder L_7	85
Figure 5.2. One-Page Embedding for L_7	86
Figure 5.3. Two-Page Embedding for L_7	86
Figure 5.4. Superhamiltonian Cycle for L_7	87
Figure 5.5. Input Representation for L_7	89
Figure 5.6. Sample G for Divide-And-Conquer	90
Figure 5.7. Results of Subproblems	91
Figure 5.8. Superhamiltonian Cycle for G	91
Figure 5.9. Exposed Vertices	93
Figure 5.10. Partition Into Subintervals	93
Figure 5.11. Results for Subintervals	94
Figure 5.12. Subintervals Strung Together	94
Figure 5.13. Parallel Edges in G	97
Figure 5.14. Removal of V_P	97
Figure 5.15. Parallel Edges	98
Figure 5.16. Supercycle for Parallel Edges	99
Figure 5.17. Replacing a Right Lower Edge	99
Figure 5.18. Replacing a Left Lower Edge	100
Figure 5.19. Adding a Subinterval on the Left	100
Figure 5.20. Adding a Subinterval on the Right	101
Figure 5.21. Proof of Lemma 5.4	109
Figure 6.1. The Genus-One Graph G_4	114

Figure 6.2. The Second Nested Triangle N_2 115

ALGORITHMS

Algorithm 3.1. Planar Graph Algorithm	50
Algorithm 4.1. Two-Page Embedding of a General Trivalent Planar Graph	74
Algorithm 4.2. Embedding a Graph with Pagenumber at Most k	74
Algorithm 4.3. Superhamiltonian Cycle of a Biconnected Trivalent Planar Graph	76
Algorithm 4.4. Oriented Face Traversal	78
Algorithm 4.5. Construction of a Subtraction Sequence	81
Algorithm 5.1. The Tradeoff Algorithm	102
Algorithm 5.2. Determining Exposed Vertices in Linear Time	103
Algorithm 5.3. Finding a Separating Edge	104
Algorithm 5.4. Generating a Maximal Set of Parallel Edges	104

CHAPTER 1

THE PROBLEM AND ITS MOTIVATIONS

1.1. The Problem

We study embeddings of graphs in structures called *books*. In this chapter, we define the book embedding problem and show that it models interesting problems in VLSI design and in parallel sorting.

A *book* consists of a *spine* and some number of *pages*. The spine of a book is a line. For simple exposition, view the spine as being horizontal. Each page of the book is a half-plane that has the spine as its boundary. Thus any half-plane is a one-page book, and a plane with a distinguished horizontal line is a two-page book.

The embedding of an undirected graph consists of two steps. The first step places the vertices of the graph on the spine in some order. The second step assigns each edge of the graph to one page of the book in such a way that on each page, the edges assigned to that page *do not cross*. Whether two edges cross is determined by the order of the vertices. If (s,t) and (u,v) are edges of the graph with $s < u < v$ and $s < t$, then the edges cross if and only if $s < u < t < v$. The resulting embedding is called a *book embedding* of the graph.

For a given graph G , there are many possible book embeddings. There are two measures of the quality of a book embedding for G . The first measure is the *pagenumber* of the embedding, which is the number of pages in the book. The *pagenumber* of the graph G is the minimum pagenumber of any book embedding of G . The *pagenumber* of a class of graphs is the minimum number of pages that every member of the class can be embedded in, as a function of graph size.

The *width* of a page is the maximum number of edges that intersect any half-line perpendicular to the spine in the page. The second measure is the *pagewidth* of the embedding which is the maximum width of any page. The *pagewidth* of the graph G is the minimum pagewidth of any book embedding of G in a book having a minimum number of pages. The *pagewidth* of a class of graphs is the minimum pagewidth that every member of the class can be embedded in, as a function of graph size. The *book embedding problem* is to find good book embeddings for a graph family with respect to one or both of these measures.

As an example, consider the grid graph G of Figure 1.1. A two-page embedding of G is shown in Figure 1.2. The vertices of G are placed on the spine in the order $A-B-C-F-E-D-G-H-I$. The first page consists of the upper half-plane, and the second page consists of the lower half-plane. Edge (B,E) of the first page crosses edge (F,I) of the second page, so these two edges cannot be assigned to the same page of this book. The pagenumber of the book

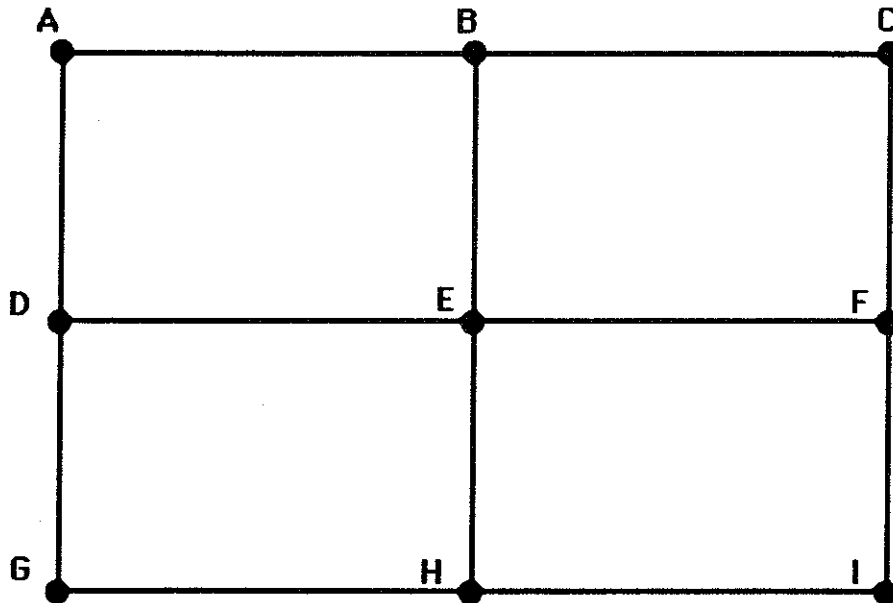


Figure 1.1. Grid Graph G

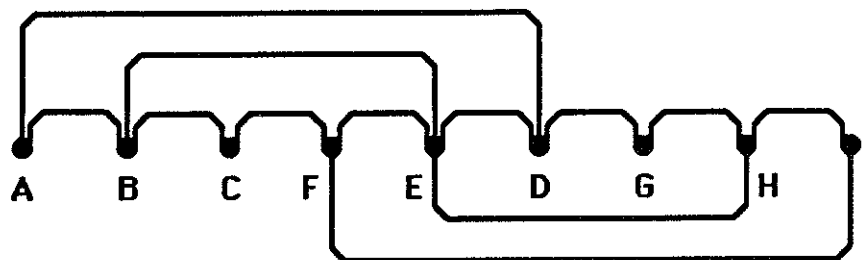


Figure 1.2. Two-Page Embedding of G

embedding is two, and the pagewidth is three as witnessed by the nested edges (A,D) , (B,E) , and (C,F) (both measures are optimal for G).

1.2. Motivations

The book embedding problem is of interest because it models problems in several areas of computer science. We mention here three particular motivating problems (see Chung, Leighton, Rosenberg [CLR] for other motivations). Our primary motivation is VLSI design; one problem is multilayer VLSI layout, and a second is design of fault-tolerant arrays of VLSI processors. The third problem is sorting with parallel (noncommunicating) stacks.

1.2.1. Multilayer VLSI Layout

VLSI layout theory has been primarily a two-layer and two-dimensional theory (Leiserson [Le]). The model of the theory is a simple and intuitively appealing one. An undirected graph represents a VLSI circuit. The vertices of the graph correspond to components of the circuit, and the edges of the graph correspond to connections or wires in the circuit. A two-dimensional grid graph represents the two-dimensional layout surface. The discreteness of the grid graph models the design rules of a VLSI technology.

The VLSI layout problem is to map (embed) the circuit graph into the grid graph. Every vertex of the circuit graph maps to a distinct vertex in the grid graph. Every edge of the circuit

graph maps to a path in the grid graph; no two paths may share an edge of the grid graph. While no two paths may share an edge, two paths may share a vertex. In the VLSI layout, one thinks of a shared vertex as a point where two wires cross. As two crossing wires must not be electrically common, the wires must be on two different layers. Two layers are sufficient for any layout; if two crossing wires are initially on the same layer, one of the wires must undergo two layer changes, one on each side of the crossing.

While two layers suffice, it is often advantageous to consider multiple (more than two) layer models for circuit layout. Multilayer printed circuit boards have long been available (So [So], Ting and Kuh [TK]). Other multilayer packaging technologies exist or are being developed (Blodgett and Barbour [BB]). Now multilayer VLSI technologies are being investigated (Locke [Lo]). The possibility of many layers takes the design problem into three dimensions. Leighton and Rosenberg [LR1] [LR2] [Ro1] [Ro2] have investigated three-dimensional layout models. Their results show that in general, the volume and wire length of good three-dimensional layouts are less than the area and wire length, respectively, of the best two-dimensional layouts of the same circuit.

Book embedding does not model arbitrary three-dimensional layouts, as the components in a book are constrained to appear on a line. The book embedding problem is one of a class of graph embedding problems called *linear arrangement* problems. In a linear arrangement problem, the vertices of a graph are ordered linearly so as to optimize some measure. One example of a linear arrangement problem is the *bandwidth problem* (Garey et al. [GGJK]); the measure to be minimized is the length of the longest edge. Another example is the *min-cut linear arrangement problem* (Gavril [Gav]); the measure (called *cutwidth*) to be minimized is the maximum number of edges intersecting any line perpendicular to the linear ordering.

Linear arrangement problems occur in VLSI layout and printed-circuit board layout. Typically, a two-dimensional layout problem is decomposed into some number of one-dimensional (linear arrangement) subproblems, and each subproblem is solved independently. The hope behind such an approach to layout is that the one-dimensional subproblems will be easier to solve

than the general two-dimensional problem. A well-known example of such a subproblem is *single row routing* (Ting, Kuh and Shirakawa [TKS], Raghavan and Sahni [RS]). Here, circuit components are given as a linearly ordered set V , and connections are given by a set of *nets*, each of which is a subset of V . A number of single row routing problems can be specified depending on such restrictions as the number of wiring layers available, the maximum cutwidth allowed any layer, and whether wires may pass between components. If wires may not pass between components, then the assignment of wires to layers in single row routing is close to the page-assignment part of book embedding. In fact, if a circuit can be realized in L layers in that single row routing problem, then the circuit can be realized in a $2L$ -page book.

Book embedding models a certain multilayer circuit layout problem: the components of the circuit are placed on a line, and all wires are routed above the line. If the graph of the circuit can be embedded in a p -page book, then the circuit can be realized in p layers. The height of the layout is proportional to the pagewidth of the book embedding; a book embedding with small pagewidth corresponds to a layout with small area. Book embedding is the one-dimensional case of decomposition into layers. Little is known about the corresponding two-dimensional case.

1.2.2. Design of Fault-Tolerant Processor Arrays

Rosenberg [Ro3] has proposed the DIOGENES approach to the design of fault-tolerant arrays of processors. The elements of the approach are sketched here. One lays out some number of identical processors in a (conceptual) line. One provides sufficiently many processors so that one expects (probabilistically) that enough good processors exist to implement the desired array.

Bundles of wires with embedded switches run parallel to the line of processors. Each bundle is capable of implementing a hardware stack of connections among processors. Each connection occurs on exactly one hardware stack (bundle). For any processor, a connection to a processor on its right is pushed on a stack; each connection to a processor on its left is popped from a stack. In this way, each connection to a good processor requires one stack operation at that processor. No stack operations occur at a bad processor. Since the state of a processor as good or bad is a binary value, a single control signal can cause the shift (push or pop) of many connections. Thus,

fault tolerance is achieved by switching in only good processors.

The desired array of processors is modeled as a *connection graph*; the vertices represent the processors, and the edges represent the desired connections between processors. The DIOGENES design problem is to determine the number of stacks and the *stackwidths* (the number of connections carried by each stack) required to implement the array of processors. In a way analogous to a hardware stack, it is possible to view one page of a book embedding as a stack of edges. For any vertex, each incident edge that connects it to a vertex to its right is pushed on a stack; each incident edge that connects it to a vertex to its left is popped from a stack. The DIOGENES design problem for an array of processors is exactly the book embedding problem for the corresponding connection graph. The number of stacks is exactly the number of pages. The stackwidths are the widths of the pages.

1.2.3. Sorting with Parallel Stacks

Even and Itai [EI], Rosenstiehl and Tarjan [RT], and Tarjan [Ta] have studied the problem of realizing a permutation with some number (say p) of noncommunicating stacks. Let π be a permutation of $\{1, \dots, n\}$. The realization of π by p parallel stacks has two stages. First, in the order $1, \dots, n$, each integer in the set is pushed on one of the p stacks. Second, in the order $\pi(1), \dots, \pi(n)$, each integer is popped from one of the p stacks. Of course, for $\pi(i)$ to be popped, it must be on the top of its stack after $\pi(1), \dots, \pi(i-1)$ have been popped. The problem can be modeled as a graph-theoretic problem. Let G be the bipartite graph with vertex set $\{u_1, \dots, u_n, v_1, \dots, v_n\}$ and edge set $\{(u_k, v_i) \mid 1 \leq k \leq n\}$. Place the vertices of G on a line in the order

$$u_1, \dots, u_n, v_{\pi(1)}, \dots, v_{\pi(n)}.$$

Then this order for G can be realized in a p -page book exactly when π can be realized with p parallel stacks ([CLR]).

1.3. Structure of the Dissertation

The dissertation contains six chapters, of which this is the first. The second chapter reviews what was known about book embeddings before our work. We prove book embedding properties for three classes of graphs in chapters three through five. Our proofs are constructive; therefore, much of the content of each of these three chapters is an efficient algorithm that constructs a book embedding with the desired property.

In the third chapter, we are interested in the pagenumber of the class of planar graphs, which we call PPG . It was already known that $3 \leq PPG$. Our first result is an algorithm that embeds any planar graph in a seven-page book. Thus, $3 \leq PPG \leq 7$. The algorithm executes in time linear in the size of the planar graph. It proves the smallest upper bound known for PPG .

In the fourth chapter, we are interested in the maximum valence for a planar graph that guarantees that it is two-page embeddable. We call this maximum valence MV . There is an easy example that shows that $MV \leq 7$. Our second result is an algorithm that embeds any trivalent planar graph in a two-page book. Thus, $3 \leq MV \leq 7$. The algorithm executes in time linear in the size of the input graph.

In the fifth chapter, we seek small pagewidth embeddings for outerplanar graphs. Let G be a d -valent outerplanar graph with n vertices. Our third algorithm embeds G in a two-page book having pagewidth less than $Cd \log n$ where $C = 8 / (\log \frac{3}{2})$ (all logarithms are to the base two). This result is within a constant factor of optimal in pagewidth for the class of outerplanar graphs. The algorithm executes in time $O(n \log n)$.

The sixth chapter sums up the significance of the work and makes suggestions for future research.

We include a glossary of graph-theoretic terms. These terms are of two kinds. First, there are new terms that we define whose use spans more than one chapter. Second, there are terms of wider use in graph theory that the reader may know by a different name. Whenever the reader encounters a term that is not defined in the current chapter, he should consult the glossary.

CHAPTER 2

PREVIOUS RESULTS AND TOOLS

2.1. Previous Results

Bernhard and Kainen [BK] is the first important work on book embeddings. They characterize one- and two-page embeddable graphs and show that a book embedding problem can be reformulated as a circular embedding problem. They show that K_n , the complete graph on n vertices, has pagewidth $\lceil n/2 \rceil$. They are the first to raise the problem of determining PPG , the pagewidth of the class of planar graphs; they make a conjecture whose truth would imply that PPG is infinite. Buss and Shor [BS] disprove this conjecture by showing that $PPG \leq 9$.

Chung, Leighton and Rosenberg [CLR] is the other major work on book embeddings. They establish the connection between book embedding and sorting with noncommunicating stacks; they use this connection to obtain the best lower bound techniques known for book embeddings. In particular, they establish a (nonconstructive) lower bound on the pagewidth of the class of d -valent graphs:

Proposition 2.1. [CLR] For $d > 2$ and sufficiently large n , there exist n -vertex d -valent graphs whose pagewidth is at least

$$(\text{constant}) \frac{n^{\frac{1}{2} - \frac{1}{d}}}{\log^2 n}.$$

Chung, Leighton and Rosenberg also develop (nonconstructive) upper bounds for d -valent graphs:

Proposition 2.2. [CLR] Let G be an n -vertex d -valent graph. Then, for all constant $\epsilon > 0$, G is $F(\epsilon, d, n)$ -pagewidth embeddable where

$$F(\epsilon, d, n) = \min\left\{\frac{n}{2}, (1+\epsilon)(2+2^{\frac{1}{2}})(d+1)n^{\frac{1}{2}}\right\}.$$

Their upper bound result for trivalent graphs is constructive:

Proposition 2.3. [CLR] Every n -vertex trivalent graph can be embedded in a $(\frac{3}{2}n^{\frac{1}{2}}+2)$ -page book with pagewidth at most $6n^{\frac{1}{2}}+8$.

Chung, Leighton and Rosenberg also present optimal (or near optimal) book embeddings for a large number of classes of graphs. Any n -node d -ary tree can be embedded in a one-page book with pagewidth at most $\frac{d}{2}\log_2 n$. An $n \times n$ grid can be embedded in a two-page book with pagewidth n . A depth- d X-tree can be embedded in a two-page book on pages of widths $2d$ and $3d$. A Boolean n -cube can be embedded in an n -page book with one page of width 2^k for $1 \leq k \leq n$. Any series-parallel graph is two-page embeddable. Building on the work of Chung, Leighton and Rosenberg, Games [Ga] shows that each of the FFT network, the Benes permutation network, and the barrel shifter network is embeddable in a three-page book (which is optimal).

Our three algorithms operate on three classes of planar graphs. The correctness of our algorithms depends only on the basic results derived by [BK] and on some basic properties of planar graph embeddings. In this section, we describe these basic results and properties. In the next section, we describe the tools that can be applied to the problem of book embedding planar graphs.

2.1.1. Circular Embedding

The original statement of book embedding is a linear embedding performed in two parts. First, the vertices of a graph are placed on a line in some order. Second, each edge of the graph is embedded in one page so that no edges in the same page cross.

The resulting linear embedding can be transformed into a circular embedding in three steps. First, choose a distinct color for each page of the book, and assign each edge the color of its page. Second, "close" the book by projecting all pages (and their edges) into a single page. In this one-page book, if two edges cross, then the two edges have different colors. Third, curve the spine into a circle so that the "ends" at infinity are identified.

The result of the transformation is an alternate two-part formulation of the book embedding problem. First, order the vertices of the graph on a circle. Second, draw the edges of the graph as chords of the circle. Color the chords (edges) so that if two chords intersect in the interior of the circle, the chords have different colors. The number of colors in the circular embedding is exactly the number of pages in the corresponding linear embedding. From now on, we shall use whichever formulation is most convenient.

A useful consequence of the circular formulation is that any p -page graph is a subgraph of a p -page hamiltonian graph. Moreover, the order of the vertices in the circular embedding is exactly the order of the vertices in the hamiltonian cycle. To see this, let v_1, v_2, \dots, v_n be the vertices of the p -page graph in the cyclic order of the circular embedding. Add each of the edges (chords) (v_k, v_{k-1}) , $1 \leq k \leq n$ (where $k-1$ is taken modulo n) that are not already present. Since these edges connect vertices adjacent on the circle, they cannot intersect any other edges. Therefore, each of the edges can legitimately be assigned to any page. The resulting edge-augmented graph is a p -page graph, with hamiltonian cycle v_1, \dots, v_n .

The idea of adding edges to a graph to obtain a hamiltonian cycle is our first tool. We will call a cycle obtained in this fashion *superhamiltonian*. The following heuristic for book embedding a graph G is proposed in [CLR]:

- (1) obtain a superhamiltonian cycle for G and place the vertices of G on the circle in the order of the cycle;
- (2) color the edges of G by coloring the associated circle graph.

Finding an optimal solution to the second step in the heuristic is an NP-complete problem (Garey et al. [GJMP]). The first step can be done in a number of ways; in fact, any ordering of the vertices can be obtained for a superhamiltonian cycle by adding the right edges. Thus, the problem of finding good book embeddings can be approached as that of finding a superhamiltonian cycle in an intelligent fashion so that a good (but not necessarily optimal) edge coloring can be produced.

2.1.2. One-Page Graphs

Any one-page graph can be embedded in the plane so that its vertices are on the spine and its edges are in the first page (the upper half-plane). Then all its vertices are exposed to the lower half-plane, which is a subset of the exterior face of the embedding. Thus the graph is outerplanar.

One characterization of an outerplanar graph is that its vertices can be embedded on a circle so that all its edges are inside the circle and no two edges intersect. This is just the condition that the graph be one-page embeddable under the circular formulation. We have the following:

Proposition 2.4. [BK] G is one-page embeddable if and only if it is outerplanar.

In fact, a k -page embedding of a graph G yields a decomposition of G into k outerplanar subgraphs, one for each page. The subgraphs share the vertices of G but are edge-disjoint. The outerplanarity of each subgraph is witnessed by the same circular ordering as that of the original book embedding.

2.1.3. Two-Page Graphs

Each two-page graph is a subgraph of a two-page hamiltonian graph. Every two-page graph is planar since the two half-planes (pages) together form a plane. Thus a two-page graph is a subgraph of a planar hamiltonian graph.

Define a graph to be *subhamiltonian* if it is the subgraph of a planar hamiltonian graph. Given a subhamiltonian graph G , it is easy to show that G has a two-page embedding ([BK]). Edge-augment G to obtain a superhamiltonian cycle in a planar graph. Order the vertices of G on a circle according to the superhamiltonian cycle. The edges of G interior to the cycle form an outerplanar graph. The edges exterior to the cycle form another outerplanar graph with its vertices in the same order as those of the interior one. A two-page embedding of G results. Thus we have the following:

Proposition 2.5. [BK] G is two-page embeddable if and only if it is subhamiltonian.

2.1.4. Planar Graphs

There are maximal (i.e., triangulated) planar graphs that are not hamiltonian. (The smallest maximal planar graph that is not hamiltonian can be found in Capobianco and Molluzzo [CM].) In Subsection 3.2.1 of Chapter 3, we shall see a way of generating a sequence of such examples. Each such example requires at least three pages in any book embedding since it cannot be edge-augmented and remain planar. Hence, three is a lower bound on *PPG*. There is no known example of a planar graph that requires more than three pages.

The problem of determining whether a planar graph is two-page embeddable is NP-complete, as witnessed by the following:

Proposition 2.1. (Wigderson [Wi]) The problem of determining whether a planar graph is the subgraph of a hamiltonian planar graph is NP-complete.

2.2. Tools

The first useful tool in reasoning about planar graphs is our ability to visualize and draw a planar graph in two dimensions. In the case of maximal planar graphs, there are additional useful properties.

Proposition 2.2. (Harary [Ha]) If a planar graph is maximal, then it is three-connected, and its planar embedding is essentially unique (i.e., unique up to the choice of the unbounded face).

In the planar graph algorithm of Chapter 3, we shall always manipulate a fixed planar embedding all of whose interior faces are triangles. It will be possible to describe the algorithm with drawings of the planar embedding.

The second useful tool is the fact that any simple, closed curve in the plane separates the plane into two disjoint regions (Jordan curve theorem). Thus the removal of any cycle in a connected planar graph separates the graph into two components (unless the cycle bounds a face). If the planar embedding is given, then we can unambiguously speak of the interior and exterior of a cycle. We can say that a vertex or edge is either *on* the cycle, *inside* the cycle or *outside* the cycle.

A third useful tool is the recognition of cycles in a planar graph. A cycle in a planar graph yields an ordering for its vertices and an assignment of edges to one of two pages. These edges are not only the edges on the cycle but are also any edges with both endpoints on the cycle. The importance of this tool will be evident as its use recurs in the planar graph algorithm. An approach to embedding a planar graph in a book is then to seek cycles in the graph, or in fact *supercycles* (a supercycle is a cycle obtained by edge-augmenting the original graph). In a general sense, this is the approach we take in each of our algorithms.

CHAPTER 3

EMBEDDING PLANAR GRAPHS IN SEVEN PAGES

Our main result in this chapter is a linear-time algorithm to embed any planar graph in a seven-page book. Along the way to obtaining the algorithm, we examine the approaches that others have taken to this problem. These results originally appeared in [He] in an abbreviated form.

3.1. Overview of the Algorithm

In this section, we introduce our planar graph algorithm with an example. The terms *levels* and *nesting* are described here intuitively; they are defined later in the chapter. The graph G used in the example is illustrated in Figure 3.1. G is a triangulated planar graph.

The algorithm partitions the vertices of G into *levels* based on distance from the cycle (u_1, u_2, u_3) bounding the exterior face. Thus, $\{u_1, u_2, u_3\}$ is level 0, $\{v_1, v_2, v_3, v_4, v_5\}$ is level 1, and $\{w_1, w_2\}$ is level 2. The algorithm recognizes cycles at each level. At level 0, (u_1, u_2, u_3) is the only cycle; at level 1, (v_1, v_2, v_3) and (v_3, v_4, v_5) are the only cycles; at level 2, there are not cycles. We see that level 2 is not even connected, and that the two components of level 2 are contained in the interior of different cycles of level 1. This is a general phenomenon; any component of level $k, k > 0$, is contained in the interior of a single cycle of level $k-1$.

The algorithm proceeds level by level, starting at level 0. It orders the vertices of each level k so that the cycles of level k are placed in cycle order. The result of embedding level 0 of G is shown in Figure 3.2. The order of vertices u_1 , u_2 , and u_3 remains the same throughout the algorithm, though vertices of succeeding levels are mingled among the level 0 vertices.

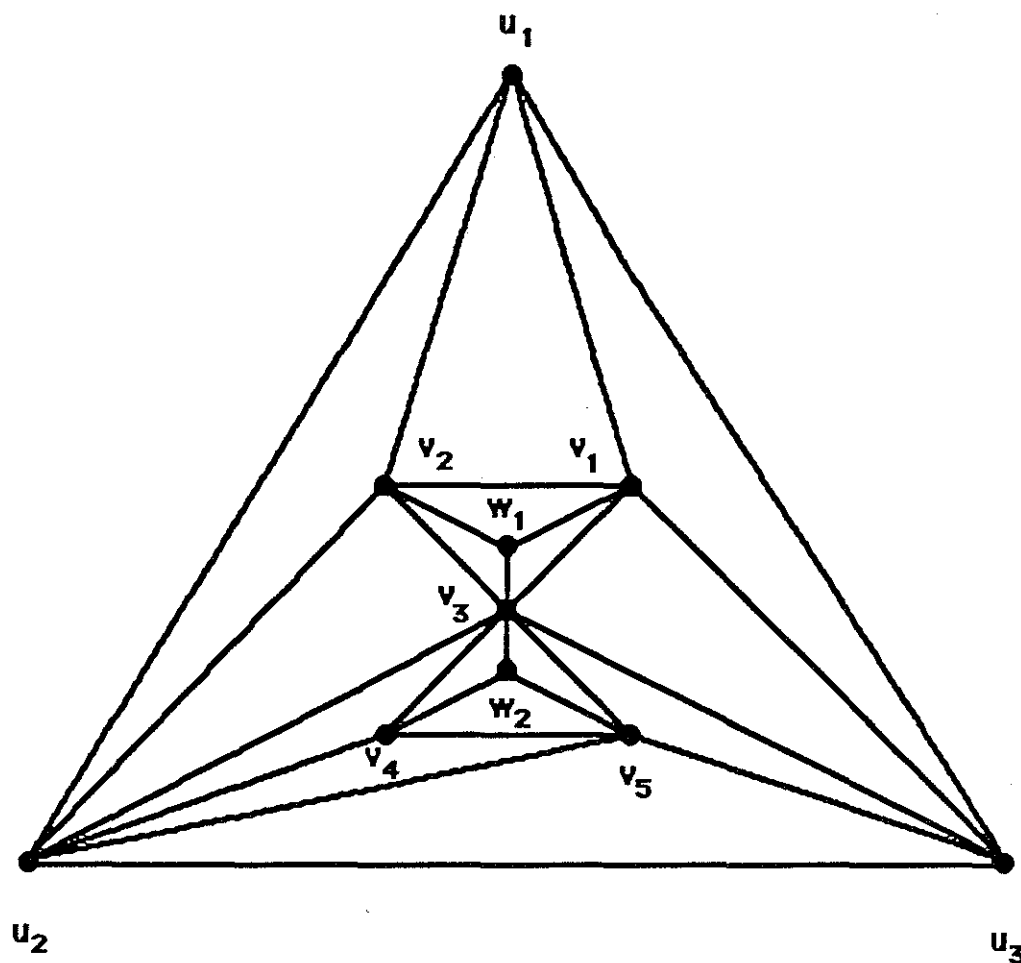


Figure 3.1. Sample Planar Graph G

In going from level 0 to level 1, the algorithm extends cycle (u_1, u_2, u_3) to a cycle that includes all level 1 vertices. As much as possible, the cycles of level 1 are placed consecutively and in cycle order in the extended cycle. In cases where two or more cycles of level k share a vertex, it may not be possible to place cycle vertices consecutively; for example, vertex v_3 is shared by both level 1 cycles. The extended cycle is $(u_1, v_1, v_3, v_2, u_2, v_4, v_5, u_3)$; it is shown with dashed lines in Figure 3.3. In extending a cycle from level 0 to level 1, vertices at levels greater than 1 are ignored; thus, the level 2 vertices of G are not drawn in Figure 3.3. Note that both level 1 cycles

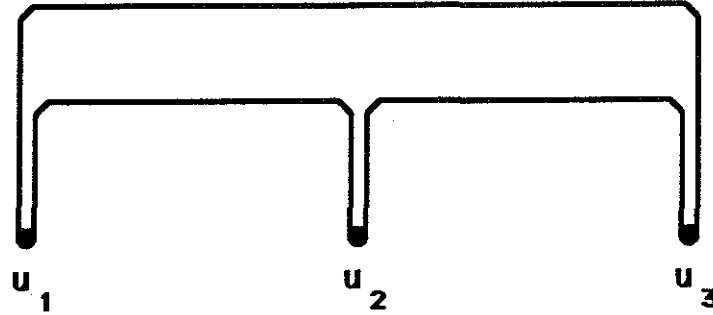


Figure 3.2. Embedding of Level 0

are in cycle order, but that the vertices of cycle (v_3, v_4, v_5) is not consecutive in the extended cycle.

The extended cycle determines the placement of the level 1 vertices with respect to the level 0 vertices; see Figure 3.4. Since the extended cycle is a hamiltonian cycle for the subgraph induced by levels 0 and 1, the edges encountered up to this point can be assigned to two pages. The edges *on* the hamiltonian cycle or *inside* the hamiltonian cycle are assigned to the upper page; edges *outside* the hamiltonian cycle are assigned to the lower page.

Level 1 has two cycles. The algorithm extends each of these cycle to include the vertices at level 2 in its interior. The two extended cycles are (v_1, w_1, v_3, v_2) and (v_3, w_2, v_4, v_5) . These determine the placement of the level 2 vertices with respect to the level 1 cycles. Figure 3.5 illustrates the placement of w_1 and w_2 , and the assignment of the edges of levels 1 and 2 to pages. Note that three pages are present in Figure 3.5: a solid upper page, a dashed upper page, and a solid lower page. Because of the vertex v_3 shared between two level 1 cycles, a third page is required.

Our example is too small to demonstrate the use of seven pages. However, we can say that at most two pages are needed to extend cycle (v_1, v_3, v_2) to include w_1 , and at most two pages are needed to extend cycle (v_3, v_4, v_5) to include w_2 . Edges from v_3 to v_4 , v_5 , and v_2 will be in a single page, so that the second page used by cycle (v_3, w_2, v_4, v_5) can be identical to one of those used by cycle (v_1, w_1, v_3, v_2) . Further page sharing is possible. The components of a level of G are *nested*

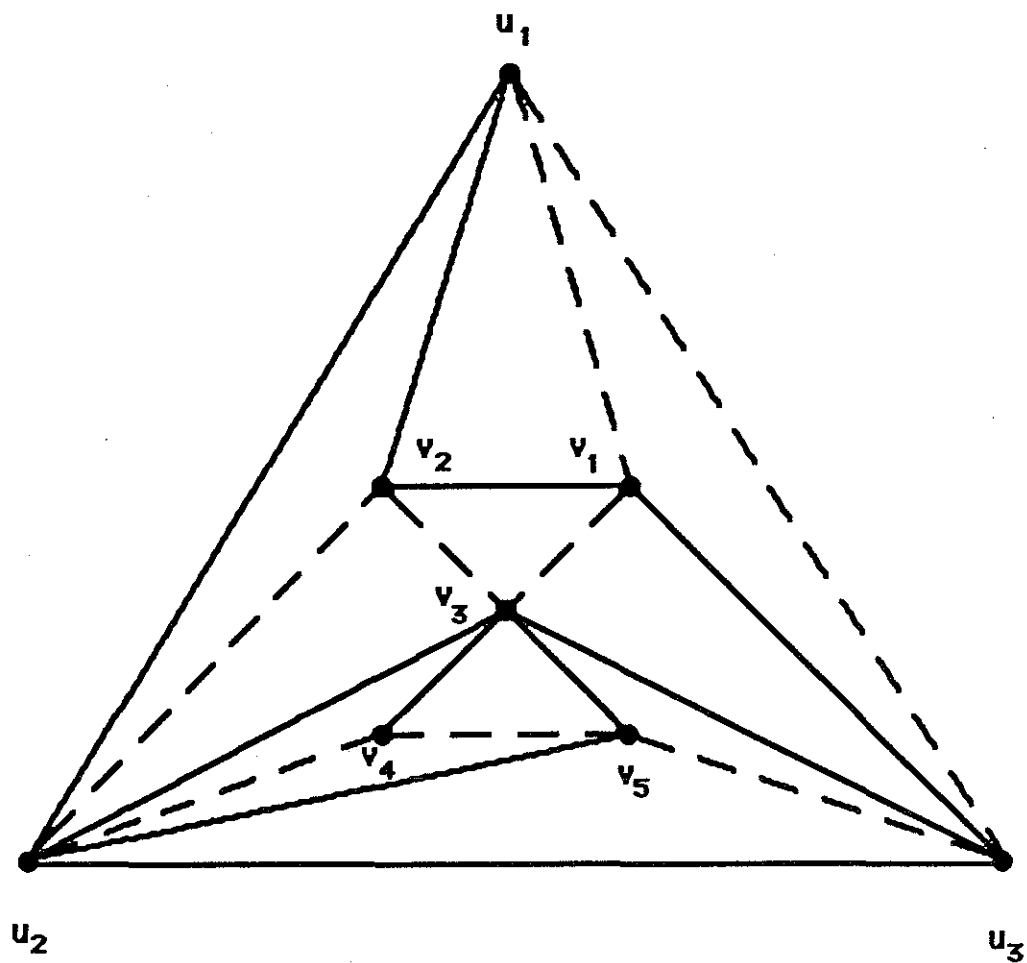


Figure 3.3. Extended Cycle

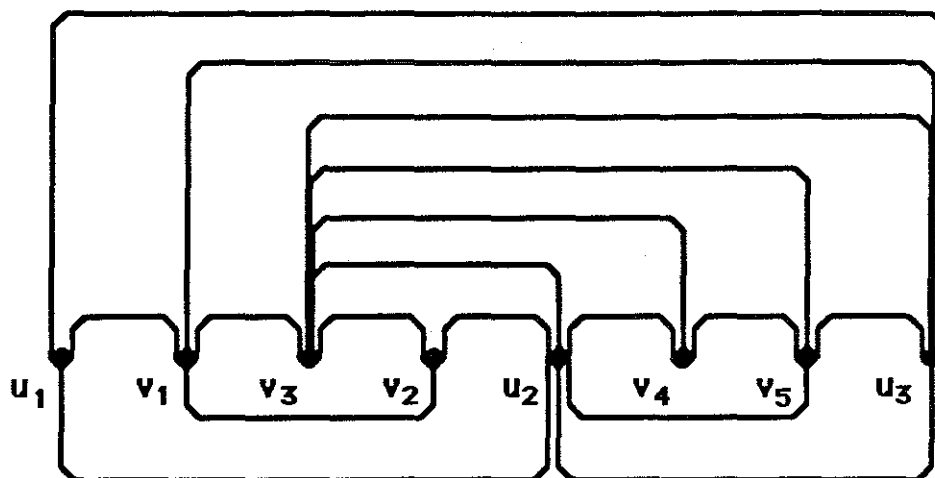


Figure 3.4. Embedding of Levels 0 And 1

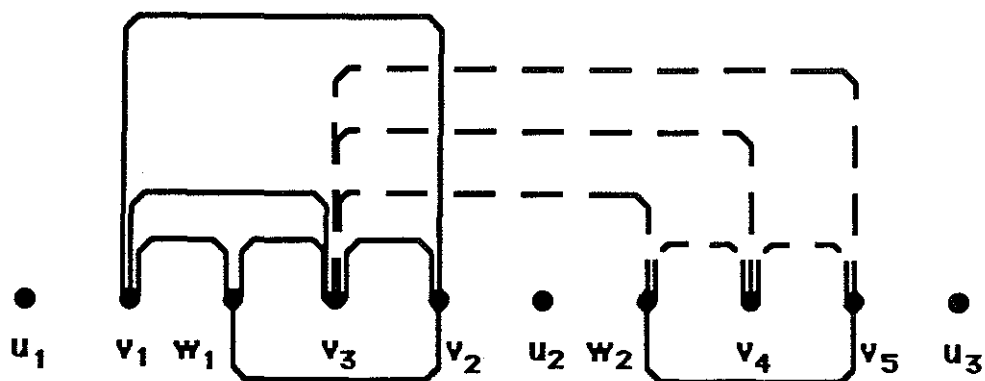


Figure 3.5. Embedding of Levels 0, 1, And 2

inside the embeddings of preceding levels. Thus, if G had more than three levels, edges at later levels would not cross edges at level 0. Thus, the first two pages can be reused. In general, pages can be reused at alternating levels.

3.2. Previous Approaches

In this section, we review other approaches that have been taken to the problem of bounding PPG . Along the way, we disprove a conjecture of Bernhart and Kainen [BK]. The truth of that conjecture would have implied that PPG is unbounded.

3.2.1. The Bernhart-Kainen Conjecture

Bernhart and Kainen [BK] first raised the question of the pagenumber of planar graphs. They recognized the importance of cycles to the book embedding problem for planar graphs. We present the reasoning that led to their conjecture that PPG is unbounded.

Let G be a maximal planar graph with n vertices. Define the *hamiltonian ratio*, $ht(G)$, to be m/n where m is the length of the longest cycle in G . If G is hamiltonian, then $ht(G)=1$. Define the *stellation* of G , $ST(G)$, to be the graph obtained by adding one vertex to each face of G and three edges from the added vertex to the three vertices on the face. $ST(G)$ is clearly planar. Define $ST^0(G)=G$ and $ST^k(G)=ST(ST^{k-1}(G))$ for all $k \geq 1$. Bernhart and Kainen showed that $ht(ST^k(G)) \rightarrow 0$ as $k \rightarrow \infty$. In other words, the longest cycles of successive stellations becomes vanishingly small with respect to the size of the stellations. Beyond some point the stellations are no longer hamiltonian. In fact, the number of edges that must be added to obtain a superhamiltonian cycle becomes large with successive stellations. This evidence led Bernhart and Kainen to conjecture that the pagenumber of $ST^k(G)$ is unbounded as $k \rightarrow \infty$.

We disprove this conjecture in the case that $G=K_3$, a triangle. In fact, we show that each $ST^k(K_3)$ is three-page embeddable. This result is best possible since $ST^2(K_3)$ is non-hamiltonian.

Let $\{a, b, c\}$ be the vertices of K_3 and $\{(a, b), (b, c), (a, c)\}$ its edges. Figure 3.6 shows $ST(K_3)$ where vertex d has been added to the interior face of K_3 and vertex e has been added to the exterior face. Figure 3.7 shows a three-page circular embedding of $ST(K_3)$. The three different pages are represented by the solid, dashed and dotted lines.

Theorem 3.1. For each $k \geq 1$, $ST^k(K_3)$ is three-page embeddable. For $k \geq 2$, this is optimal.

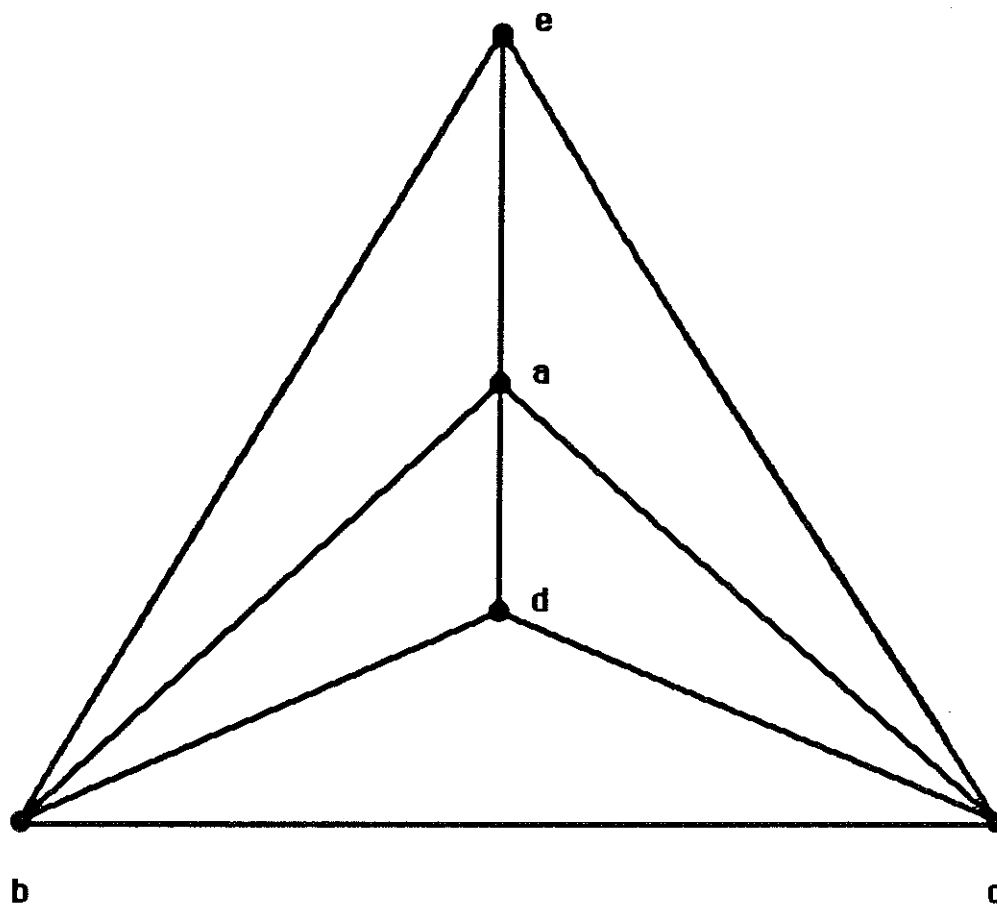


Figure 3.6. $ST(K_3)$

Proof: The proof is by induction on k . It turns out that the embedding of the stellations exterior to K_3 are symmetric to those interior to K_3 . Thus we restrict attention to the interior stellations.

We need some additional conditions on each three-page embedding to make the induction go through. Suppose z is the vertex added to the face (u, v, w) of $ST^{k-2}(K_3)$ on the way to obtaining $ST^{k-1}(K_3)$. We need the following conditions to hold on the embedding of $ST^{k-1}(K_3)$:

- (1) for one of u, v or w (say u), the arc in the embedding from u to z that *avoids* (i.e. does not contain) v and w also avoids all other vertices of $ST^{k-1}(K_3)$;

(2) the three added edges (u,z) , (v,z) and (w,z) are embedded in three distinct pages.

Taking vertex d in Figure 3.7 to be z , condition (1) is met by arc cd . Condition (2) is satisfied for d since its three edges are on three distinct pages. Thus we have the embedding we need for the induction when $k=2$.

So assume that $k>2$, and that we have a three-page embedding of $ST^{k-1}(K_3)$ satisfying conditions (1) and (2). Suppose z, u, v and w are as in the preceding paragraph. Figure 3.8 shows this situation, where we know there are no vertices on the solid arc uz but there may be vertices on the dotted arcs. Assume vertex r is added to face (u,v,z) , vertex s is added to face (u,w,z) and vertex t is added to face (v,w,z) . The vertices r, s, t and their incident edges are embedded as in Figure 3.9. Vertices r and t are placed in the arc uz in the order $urtz$. Vertex s is placed on the opposite side of z and close enough to z so that there are no vertices in the arc zs .

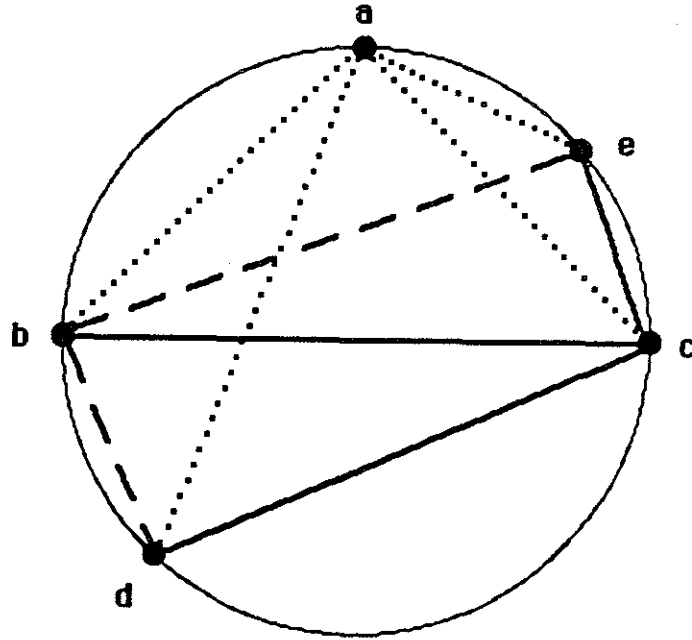


Figure 3.7. Three-Page Embedding of $ST(K_3)$

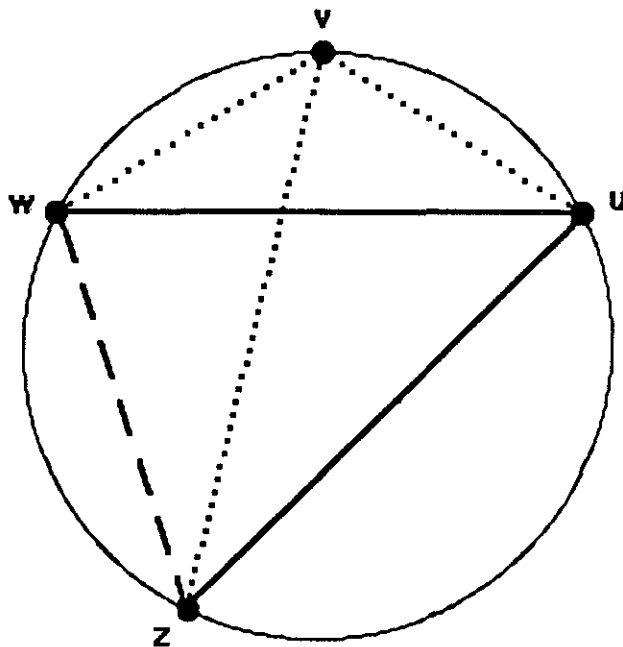


Figure 3.8. Inductive Hypothesis for Theorem 3.1

We must prove that the added edges can be assigned to the pages shown. The edges (u,r) , (t,z) and (s,z) can be on any page since each pair of vertices is adjacent. Edge (r,z) can be assigned to the solid page since the only edges it intersects are (t,v) and (t,w) , neither of which can be on the solid page (each intersects the solid edge (u,z)). The page assignment of each of the remaining added edges can be justified by its being "protected" by another edge in the same page. For example, (s,w) can be assigned to the dashed page because edge (w,z) is dashed. By construction, there is no vertex in the arc zs . Thus any edge that intersects (s,w) also intersects (w,z) . Any such edge cannot be on the dashed page. A similar protection argument justifies the page assignment for the remaining edges.

Now conditions (1) and (2) are satisfied for the added vertices r , s and t . For example, vertex r was added to face (u,v,z) . There are no vertices in the arc u , and the three edges incident to r are in three distinct pages. The theorem now follows by induction. \square

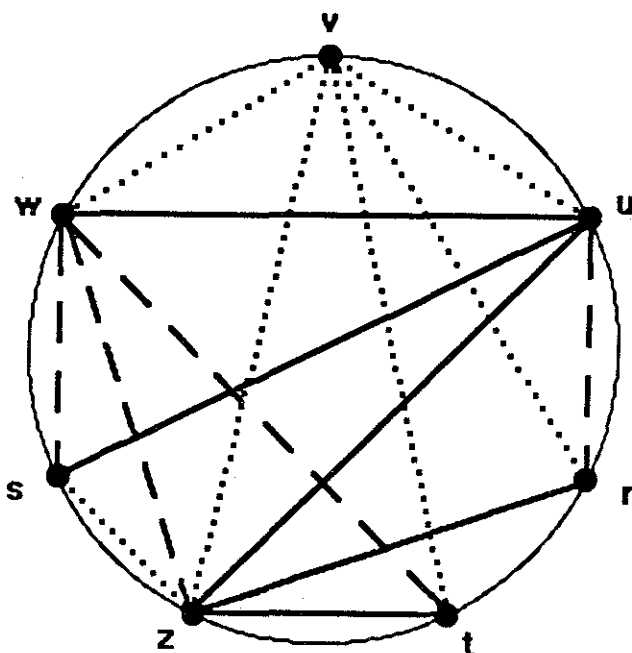


Figure 3.9. Inductive Step—Add r, s And t

As an example, Figure 3.10 shows the interior of $ST^2(K_3)$. Figure 3.11 gives the three page circular embedding that results from the construction of the theorem.

Theorem 3.1 can be generalized a bit. Suppose there exists a three-page embedding of $ST(G)$ that satisfies conditions (1) and (2) in the proof of the theorem. Then $ST^k(G)$ is three-page embeddable for all k . The proof is the same as for the theorem.

Since an arbitrary planar graph is not a stellation of some three-page graph, we cannot hope to generalize the theorem to the class of all planar graphs. However, we can abstract the general approach. Call the vertices added to $ST^{k-1}(G)$ to obtain $ST^k(G)$ the *level k vertices*. The original vertices of G are the *level 0 vertices*. Start with the level 0 vertices placed on a circle and the remainder of $ST^k(G)$ in the interior of the circle. For each successive level j , $1 \leq j \leq k$, pull each level j vertex out of the interior to a place on the circle so that it is near some vertex it is adjacent to on the preceding levels. Of course, this abstraction is too rough to form an algorithm.

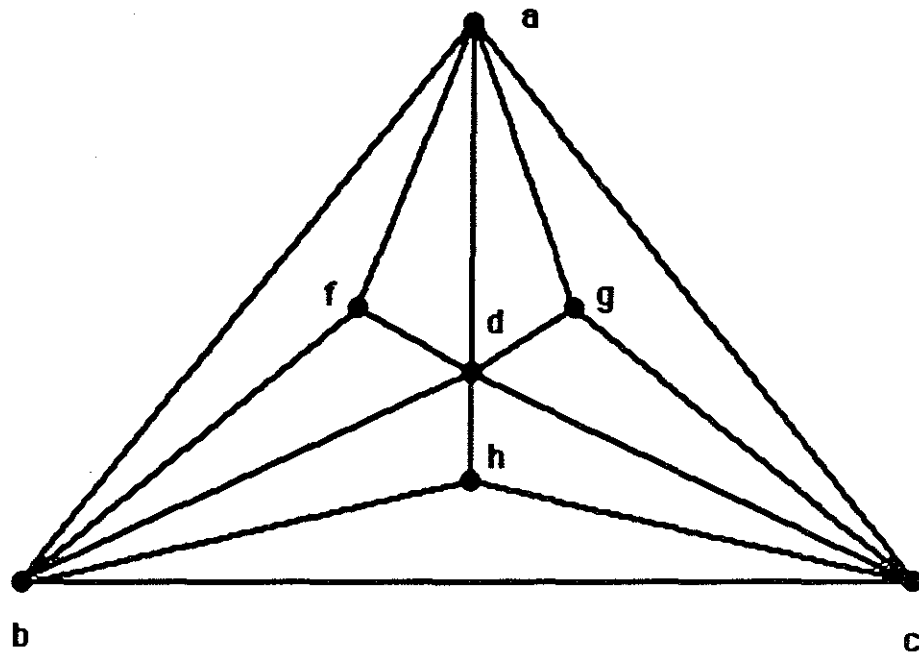


Figure 3.10. Interior of $ST^2(K_3)$

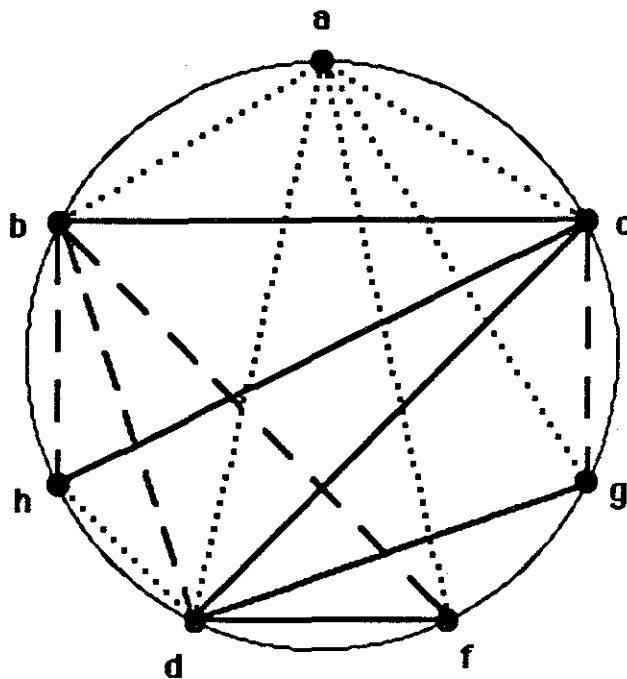


Figure 3.11. Book Embedding of $ST^2(K_3)$

However, our planar graph algorithm will, in some sense, match this abstraction.

3.2.2. Bifurcators

Chung, Leighton and Rosenberg [CLR] apply the notion of a bifurcator to obtain non-trivial upper bounds on page number for some classes of graphs. A bifurcator provides a measure of the difficulty of recursively dividing a graph. Formally, let G be a graph, B an integer and ρ a number, $\rho > 1$. We say that G has a *balanced ρ -bifurcator of size B* if G has fewer than B edges or if there exists a *decomposition tree* for G as follows. The root of the tree (at level 0) is G . Suppose H is a graph at level k . If $|H|=1$, then H has no sons. If $|H|=n > 1$, then H has two sons H_1 and H_2 at level $k+1$ such that: (a) $|H_1| = \lceil n/2 \rceil$, $|H_2| = \lfloor n/2 \rfloor$; (b) $H = H_1 \cup H_2$; and (c) the number of edges between H_1 and H_2 is no more than $B\rho^{-k}$.

The following proposition is proved in [CLR]:

Proposition 3.2. [CLR] If G has a balanced ρ -bifurcator of size B , then it is embeddable in $\left(\frac{\rho}{\rho-1}\right)B$ pages.

Each bounded-degree n -vertex planar graph has a balanced $\sqrt{2}$ -bifurcator of size $O(\sqrt{n})$. Thus, for any fixed degree d , the d -valent n -vertex planar graphs can be embedded in $O(\sqrt{n})$ pages. This approach is applicable to many classes of graphs but it does not make use of the special structure of planar graphs.

3.2.3. Separating Triangles

Buss and Shor [BS] combine the notions of hamiltonian cycles and separating cycles to yield the first proof that the pagenumber of planar graphs is bounded. In their result, the separating cycles are always triangles that are not the boundary of any face. Their approach depends on the following powerful result of Whitney [Wh]:

Proposition 3.3. [Wh] If G is a maximal planar graph with no separating triangles, then G is hamiltonian.

We briefly describe their construction. G is partitioned into successive *levels*. Each level consists of some number of connected *sections*, each of which contains no separating triangle. In fact, the separating triangles of the original G are the windows through which sections at successive levels view each other. Each section is hamiltonian by Whitney's result and hence two-page embeddable. In fact, each section together with the *subsections* at the next level that can be seen through the windows in the section can be embedded in six pages. Each subsection has at most three pages incident to it. The embedding of the subsection nests within the embedding of the section. Thus six pages are required to go between successive levels and three pages can be reused. Hence, a nine-page book embedding results.

There are three parallels between the Buss and Shor construction and our algorithm. First, both embeddings are done by levels. Second, cycles are used in some way to separate levels. Third, the embeddings of successive levels nest so that pages can be reused. We elaborate on these points in the next section.

3.3. Elements of the Seven-Page Algorithm

In this section, we discuss the significant elements of our solution while avoiding the details. We first describe a decomposition of a planar graph into *levels* in preparation for a breadth-first traversal of the levels. The argument is inductive; once all preceding levels have been embedded, with the inductive hypothesis met, the algorithm shows how to add one more level and make the inductive hypothesis true for the new embedding. *Cycles* at each level are the windows to the next level and the basis for extension to the next level.

3.3.1. Levels

The first element of our solution is to partition the vertices of a planar graph into levels. One characteristic of these levels is that each edge of the graph is either between two vertices at the same level or between two vertices at successive levels. In contrast to the levels of Buss and Shor, our levels are based on distance from the exterior face of the planar embedding. Buss and Shor obtain hamiltonian cycles in each of their levels but have no information concerning the structure of these cycles. With our leveling, we have more control on the superhamiltonian cycles obtained and have important information about the order of vertices in the cycles.

For convenience in defining levels, we restrict attention to a subclass of the planar graphs. An *inner-triangulated planar graph* (or *I-graph*) is a connected undirected simple graph that can be embedded in the plane so that any interior face is bounded by a triangle, and the exterior face is bounded by a cycle (the *bounding cycle* of the I-graph). Henceforth, whenever we have an inner-triangulated planar graph, we also have a fixed planar embedding of the above form. The *interior* and *exterior* of any cycle are defined with respect to this embedding. Clearly, any planar graph G is a subgraph of some I-graph G' . Since a seven-page embedding of the I-graph G' restricts to a seven-page embedding of the graph G , we are justified in considering only I-graphs. The introduction of I-graphs is also justified by the needs of an inductive proof. Given any cycle in an I-graph, that cycle together with its interior forms an I-graph. The algorithm will use the discovery of the bounding cycles of I-graphs at each level to continue the induction.

Let $G=(V,E)$ be an I-graph. $G_0=(V_0,E_0)$ is the *level 0 subgraph* of G where V_0 , the set of *level 0 vertices*, is the set of vertices on the exterior face of G , and E_0 , the set of *level 0 edges*, is the set of edges on the exterior face of G . Hence, G_0 is just the bounding cycle of the exterior face of G .

Levels $k>0$ are defined by distance from G_0 . Suppose we have defined $G_{k-1}=(V_{k-1}, E_{k-1})$, for $k \geq 1$. $G_k=(V_k, E_k)$, the *level k subgraph* of G , is defined as follows. V_k , the set of *level k vertices*, is the subset of $V - \bigcup_{j=0}^{k-1} V_j$, consisting of vertices adjacent to vertices in V_{k-1} . An edge (v_1, v_2) is in E_k , the set of *level k edges*, if $v_1, v_2 \in V_k$ and there exists $v_3 \in V_{k-1}$ such that (v_1, v_2, v_3) is a face of G . Clearly, V is the disjoint union of all nonempty V_k , $k \geq 0$.

X_k , the set of *level k chordal edges*, contains exactly the edges between level k vertices that are not in E_k . $B_{k,k+1}$, the set of *level k to $k+1$ binding edges*, contains an edge (v_1, v_2) if v_1 is a level k vertex and v_2 is a level $k+1$ vertex or vice versa. Clearly, E is the disjoint union of all the nonempty E_k , X_k and $B_{k,k+1}$, for $k \geq 0$.

The set E_k can be further partitioned. C_k , the set of *level k cycle edges*, is the set of those edges in E_k that lie on a cycle of G_k . $N_k = E_k - C_k$ is the set of *level k non-cycle edges*.

If K is a level k cycle, then call K together with its interior $G|K$ (read *G restricted to K*). $G|K$ is an I-graph. By the independence of the interior and exterior of K , it makes sense to speak of the levels of G not preceding k restricted to K . For example, $V_j|K$, $j \geq k$, consists of all vertices of V_j which are on K (if $j=k$) or interior to K (if $j>k$). $B_{j,j+1}|K$, $j \geq k$, consists of all edges of $B_{j,j+1}$ which are interior to K .

Let v_q be a vertex of a cycle K of G_k . Let v_{q-1} and v_{q+1} be the vertices of K adjacent to v_q (when K is traversed clockwise). Then there is a path in $G|K$ from v_{q-1} to v_{q+1} that includes only vertices adjacent to v_q , since the interior of K is triangulated. Define P_{v_q} to be this path.

Figure 3.12 shows an example of an inner-triangulated graph with two levels. Vertices v_1, \dots, v_{12} are in V_0 , while vertices u_1, \dots, u_6 are in V_1 . Edges (v_2, v_4) , (v_4, v_{10}) , (v_8, v_9) and (v_7, v_9) are level 0 chordal edges. $E_1 = \{(u_1, u_2), (u_2, u_3), (u_4, u_5)\}$. Since G_1 has no cycles, $N_1 = E_1$ and $K_1 = \emptyset$.

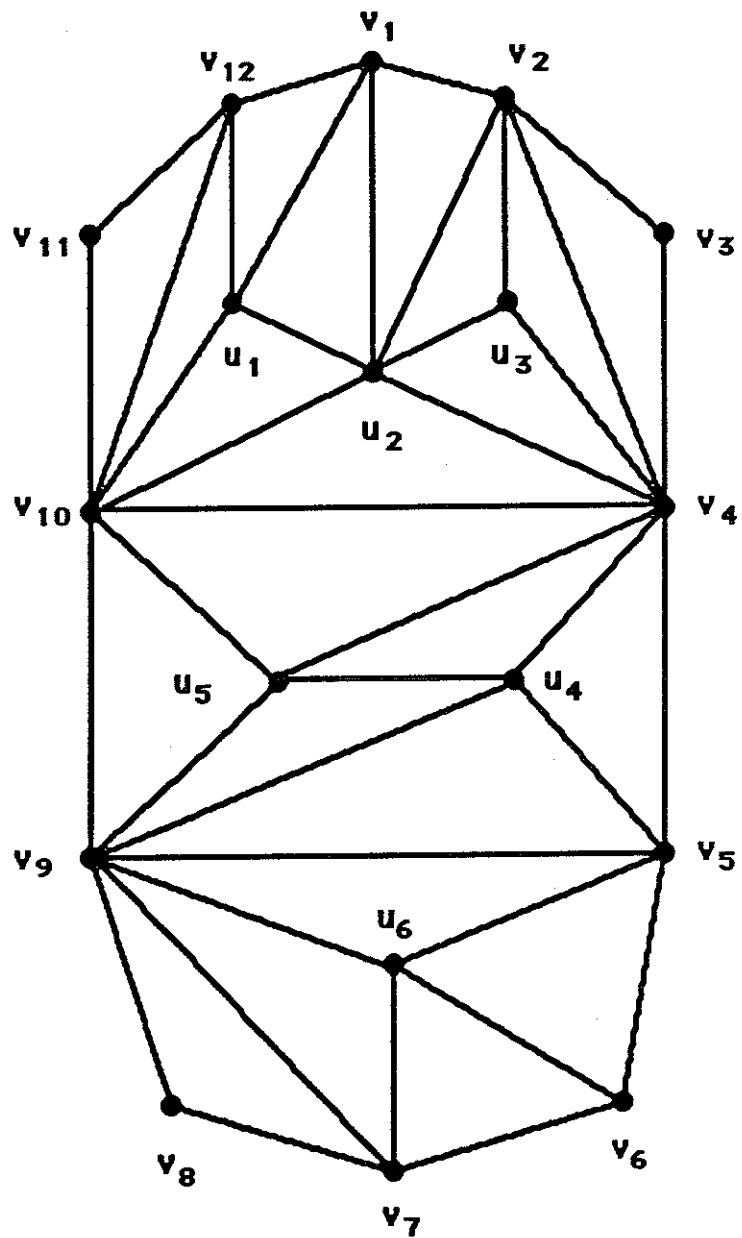


Figure 3.12. I-graph with Two Levels

Examples of level 0 to level 1 binding edges are (v_1, u_1) , (v_4, u_5) and (v_6, u_6) . P_{v_1} is the path v_{12}, u_1, u_2, v_2 .

Figure 3.13 shows an inner-triangulated graph with 3 levels. Level 0 vertices are v_1, \dots, v_6 , level 1 vertices are u_1, \dots, u_7 and the sole level 2 vertex is w_1 . The two cycles of G_1 are $(u_1, u_2, u_3, u_4, u_5)$ and (u_4, u_6, u_7) . The only chordal edge is (u_3, u_5) .

We now examine the structure of levels in an I-graph. As we noted earlier, I-graphs recur at successive levels. We study the structure of G_0 together with G_1 to understand the structure of G_k and G_{k+1} . The vertices of G_1 are just those adjacent to vertices of G_0 but not in G_0 . Remove the vertices of G_0 and their incident edges from the planar embedding. Then V_1 (the vertices of G_1) are exactly those on the exterior face of the resulting planar embedding. By the definition of E_1 and the fact that G is inner-triangulated, the edges on the exterior face are exactly E_1 . Thus we have the following:

Lemma 3.4. $G_1=(V_1, E_1)$ is an outerplanar graph.

We ask when G_1 is connected. The following lemma provides a sufficient condition for G_1 to be connected.

Lemma 3.5. If X_0 is empty, then G_1 is connected.

Proof: Let v_1, \dots, v_m be the vertices of G_0 in clockwise order. Let $u_1, u_2 \in V_1$ be such that u_1 is adjacent to v_1 and u_2 is adjacent to v_2 . We will show that u_1 and u_2 are in the same connected component of G_1 . Since the interior of G is triangulated, there exists $z \in G_1$ such that (v_1, v_2, z) is a triangle. Also, we may represent P_{v_1} by the path $(v_m, w_1, \dots, w_b, v_2)$. Similarly, there exists a path $(v_1, x_1, \dots, x_n, v_2)$ such that each vertex in the path is adjacent to v_2 . Since X_0 is empty, the path (w_1, \dots, w_i) is in G_1 . Similarly, the path (x_1, \dots, x_k) is in G_1 . Clearly, by planarity and triangulation, $z=w_i=x_1$. Since u_1 is some w_n , by triangulation, u_1 and z are in the same connected component of G_1 . Similarly, u_2 and z are in the same component. Hence, u_1 and u_2 are in the same component. By a transitivity argument, all vertices of G_1 are in a single connected component. \square

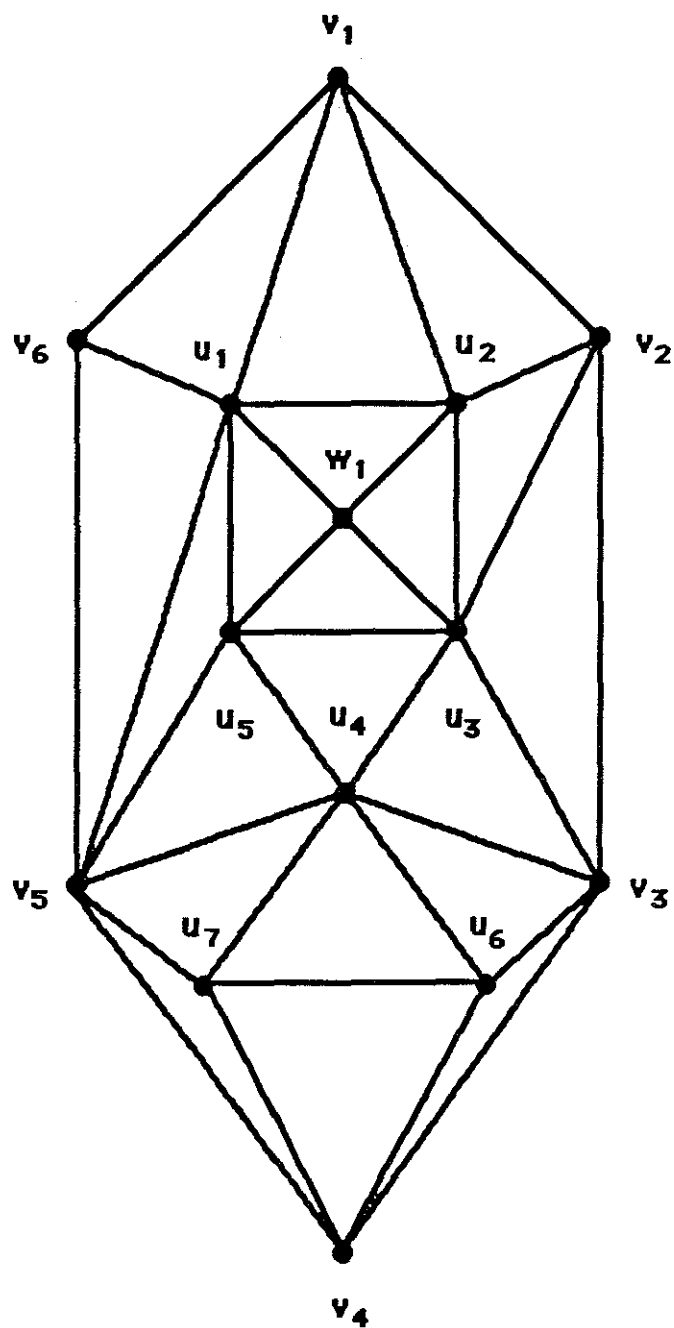


Figure 3.13. I-Graph with Three Levels

See Figure 3.14 for an example where X_0 is empty.

Assume G_1 is connected. Every biconnected component of G_1 that is not a single edge is a cycle, all of whose edges are on the exterior face (Harary [Ha]). For each cycle K of G_1 , add a vertex v_K in the interior of K , remove all the edges of K , and connect v_K to each vertex of K . Let H be the resulting graph and call it the *biconnected components graph* (or *BC-graph*) of G_1 . As an example of a connected G_1 , see Figure 3.15; G_1 contains cycles A and A' . The construction of the BC-graph adds vertex v_A in the interior of cycle A and vertex $v_{A'}$ in the interior of cycle A' . The construction then removes edges of A and A' . Finally, the construction connects v_A to all vertices of A and $v_{A'}$ to all vertices of A' . Figure 3.16 illustrates the resulting BC-graph. The construction of the BC-graph is quite similar to that of the block-cutpoint-tree of Harary and Palmer [HP].

Clearly, H is planar and connected, since G_1 is. In fact, H is a tree.

Lemma 3.5. Suppose G_1 is connected. Then H , the BC-graph of G_1 , is a tree.

Proof: By induction on the number of cycles in G_1 . If G_1 has no cycles, then G_1 is a tree and $H=G_1$. Suppose G_1 has exactly one cycle K . If $G_1=K$, then H is clearly a tree. If $G_1 \neq K$, then

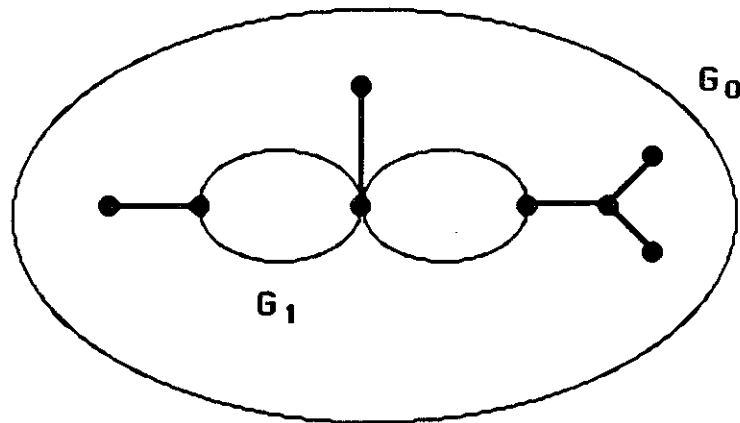


Figure 3.14. Empty X_0

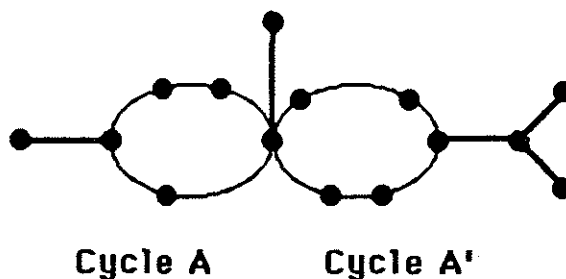


Figure 3.15. Example of Connected G_1

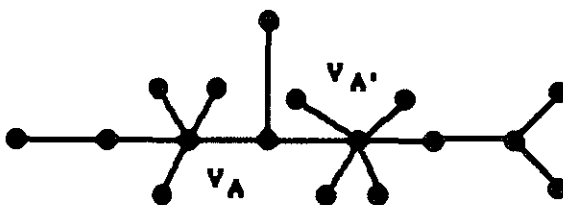


Figure 3.16. The BC-Graph of G_1

G_1 minus the edges of K is a forest where each tree in the forest contains exactly one vertex of K . The addition of v_K together with edges to each vertex thus makes the forest into a tree. Hence H is that tree.

For purpose of induction, suppose the result is true whenever G_1 has fewer than k cycles, $k > 1$. To extend the induction, assume G_1 has k cycles. Let K and A be distinct cycles of G_1 . There exists a cut vertex $v_j \in V_1$ on K whose removal separates G_1 into two or more connected

components such that K and A are in different components. Let D_1 be the component containing $K-v_j$, and let D_2 be the component containing $A-v_j$. Let D_3, \dots, D_m be the remaining components of G_1 . Each component has some vertex adjacent to v_j . By inductive hypothesis, the BC-graph of each component is a tree. Further H is exactly the union of these BC-graphs together with an edge from each BC-graph to v_j . Thus H is a tree. \square

We are now justified in calling H the biconnected components tree (or BC-tree) of G_1 when G_1 is connected. Now that we know the structure of G_1 , let us examine the connections between G_0 and G_1 . We say an edge (v_1, v_2) of G is *visible* from vertex v_3 if (v_1, v_2, v_3) is a face of G . The following lemma shows that we can characterize an edge of E_1 as a cycle edge or a non-cycle edge according to whether the edge is visible from one or two vertices of G_0 .

Lemma 3.7. Suppose $(v_1, v_2) \in E_1$. Then $(v_1, v_2) \in N_1$ if and only if there exist exactly two distinct vertices $v_3, v_4 \in V_0$ such that (v_1, v_2) is visible from v_3 and v_4 .

Proof: Suppose $(v_1, v_2) \in N_1$. By definition of N_1 , (v_1, v_2) does not lie on a cycle of G_1 . By definition of E_1 , there exists $v_3 \in V_0$ such that (v_1, v_2) is visible from v_3 . Let $v_4 \in V$ be such that (v_1, v_2) is also visible from v_4 and $v_4 \neq v_3$. If $v_4 \in V_2$, v_4 is in the interior of some cycle K of G_1 . But then (v_1, v_2) must be an edge of K , so $(v_1, v_2) \notin N_1$. If $v_4 \in V_1$, then either (v_1, v_2, v_4) is a cycle of G_1 (a contradiction) or one of (v_1, v_4) and (v_2, v_4) is in X_1 , say (v_1, v_4) . The edge (v_1, v_4) is in the interior of some cycle K of G_1 . But then (v_1, v_2) must be an edge of K , a contradiction. By process of elimination, $v_4 \in V_0$.

Now suppose there exist distinct $v_3, v_4 \in V_0$ such that (v_1, v_2) is visible from v_3 and v_4 . The edge (v_1, v_2) is in the interior of G_0 . No cycle containing (v_1, v_2) can exist in the interior of G_0 by planarity. Hence $(v_1, v_2) \in N_1$. \square

It is clear that the vertices and edges at levels ≥ 2 are contained in the interiors of the cycles of G_1 (the level 1 cycles). Since cycles in a planar graph separate the graph, the interiors of any two level 1 cycles are independent in the sense that there are no edges or vertices in common between the two interiors, and there are no edges between the two interiors.

Since $G|K$ is an l-graph, we can translate results for G_0 to results for $G|K$. In particular, we immediately have these generalizations of Lemmas 3.5-3.7.

Lemma 3.8. Suppose K is a cycle of G_{k-1} , and $X_{k-1}|K = \emptyset$. Then $G_k|K$ is connected.

Lemma 3.9. Suppose K is a cycle of G_{k-1} , and $G_k|K$ is connected. Then H , the BC-graph of $G_k|K$, is a tree.

Lemma 3.10. Suppose K is a cycle of G_{k-1} , and $(v_1, v_2) \in E_k|K$. Then $(v_1, v_2) \in N_k|K$ if and only if there exist exactly two distinct vertices $v_3, v_4 \in V_{k-1}|K$ such that (v_1, v_2) is visible from v_3 and v_4 .

We complete the description of the leveled structure of G by giving a tree decomposition for G . First define a two-level tree for the structure of G_0 and G_1 . Let K_1, K_2, \dots, K_m be the cycles of G_1 , and let V_1' be the vertices of G_1 not on any of K_1, K_2, \dots, K_m . Represent G_0 and G_1 by the non-oriented tree with G_0 as root and V_1', K_1, \dots, K_m as leaves. Now, each $G|K_j$ is an l-graph with fewer levels than G . Thus each leaf K_j can be further decomposed until a tree results where no leaf is a cycle and each non-leaf is a cycle. Call the resulting tree the *decomposition tree* (or *D-tree*) for G , and call each interior node a *D-cycle*.

The D-tree of G is almost a partition of the vertices of G . It fails to be a partition exactly in the case where two *brother* cycles at the same level share a vertex. Such a shared vertex is called a *pinch vertex*. Two brother cycles can share at most one pinch vertex. The pinch vertices at a level will be a particular source of problems to our algorithm.

The general flow of the algorithm can be described with respect to the D-tree. The vertices are ordered, and the edges are assigned to pages via a breadth-first traversal of the D-tree starting at the root. If K_i and K_j are two brother cycles, the algorithm assigns two disjoint blocks on the embedding circle to the two cycles. The important step in the algorithm is thus extending the layout of a cycle in the D-tree to a layout of the cycle and all its sons. We discuss this step next.

3.3.2. D-Cycles

The second element of our solution is the recognition of D-cycles. If a D-cycle is at level k of the D-tree, we call it a (k) -cycle. Each D-cycle separates G into two independent parts, its

interior and its exterior. Consider a particular D-cycle K . Let $K_I = G|K$ denote the graph consisting of K together with its interior. Let V_I denote the vertex set for K_I . Similarly, let K_E denote the graph consisting of K together with its exterior, and let V_E denote its vertex set. Suppose we have book embeddings B_I and B_E for K_I and K_E respectively. How can we obtain a book embedding B for G from B_I and B_E ? We recognize two points here.

First, K_I and K_E have exactly the cycle K in common. If the vertices of K do not appear in the same (cyclic) order in both B_I and B_E , then there is little hope of binding the two together. If the order of K is the same in both, then at least $V = V_I \cup V_E$ can be placed in an order such that both V_I and V_E are in the same order that they appear in B_I and B_E respectively. What common order should be chosen for K in both book embeddings? We choose the cyclic order generated by the cycle K . A corollary of this choice is the following. When embedding the exterior of K , the interior of K is not examined: the embeddings of K_I and K_E are independent except for their interface, K .

Second, if we do combine B_I and B_E to obtain an embedding B for G , the number of pages in B could be as large as the sum of the numbers of pages in B_I and B_E . This is undesirable, as we want a constant (seven) upper bound on the pagenumbers of B_I , B_E , and B . We constrain the structure of B_I and B_E so that any new crossing edges created by the combining of B_I and B_E to form B are edges incident to vertices of K . The constraint, called *nesting*, is covered in the next section.

Our algorithm constructs B by induction on the level of K . We can describe in general terms the key inductive step of extending an embedding for a D-cycle K at level $k-1$ to an embedding for K together with its sons in the D-tree. For convenience of exposition, we assume $X_{k-1}|K = \emptyset$, so that $G_k|K$ is connected, by Lemma 3.8. The removal of this assumption is discussed in the next section. By a suitable inductive hypothesis, K is embedded in its cycle order within a book embedding for K_E . We extend K to be a supercycle including all vertices of $G_k|K$, that is, all the vertices of the sons of K in the D-tree. K remains in its cycle order within the supercycle. Each D-cycle in $G_k|K$ is also in its cycle order to satisfy our requirement that the

embedding of each D-cycle be suitable as an interface to the next level.

We first describe a method of accomplishing the extension of K , that is not the one we use. Our reason for mentioning this method is twofold. First, the method does work in the case that $G_k|K$ contains no cycles (see section 3.4). Second, analysis of the failure of this method motivates the method that we do use.

The method visits each vertex of $G_k|K$. Each vertex of $G_k|K$ is adjacent to at least one vertex of K . Suppose v_q is a vertex of K . Let P_{v_q} be the path $(v_{q-1}, u_1, u_2, \dots, u_m, v_{q+1})$. Then $(u_1, u_2), (u_2, u_3), \dots, (u_{m-1}, u_m)$ are the edges of $G_k|K$ visible from v_q . In fact, P_{v_q} gives the visitation order for these edges; see Figure 3.17. The vertices of K are visited in counter-clockwise order. When vertex v_q of K is reached, the vertices u_1, \dots, u_m in P_{v_q} are examined in a clockwise order until an unvisited vertex is encountered. (If there is no unvisited vertex, then go to v_{q+1} .) If vertex u_j is unvisited, it means that u_j is not yet in the supercycle. Go from v_q to u_j and follow

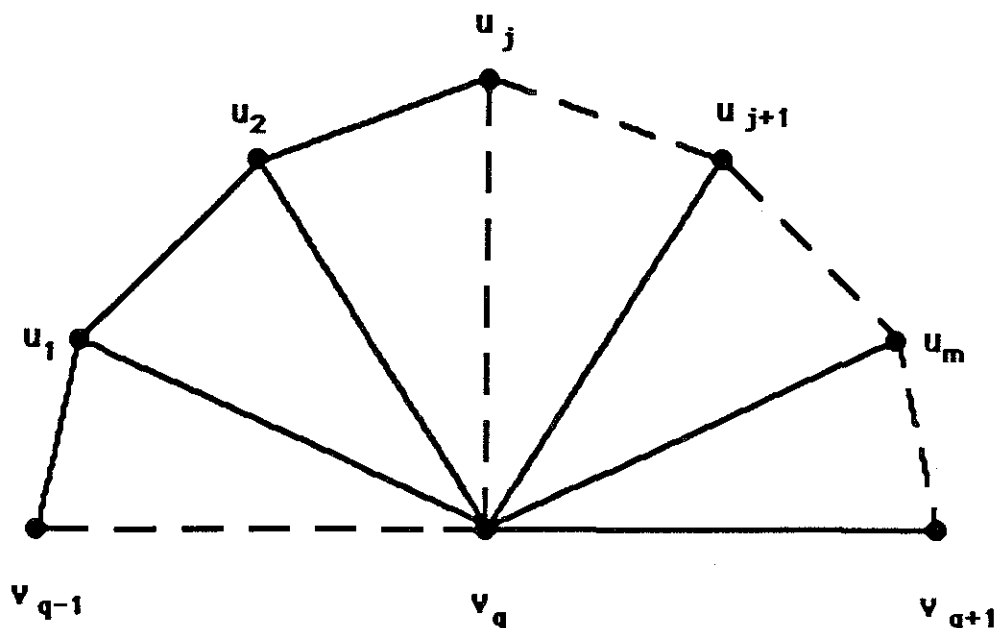


Figure 3.17 Visiting the Vertices of P_{v_q}

P_{v_i} from u_i to v_{i+1} . Once all vertices of K have been visited, a supercycle results that contains all vertices of K and $G_k | K$.

We note some features of the resulting supercycle. First, we have the reason for considering this method:

The supercycle indicates a two-page embedding for the graph consisting of K , $B_{k-1,k} | K$ and $G_k | K$; it orders the vertices and partitions the edges into two pages.

Second, this supercycle is really just a cycle in the graph; no edges are added to obtain the supercycle. Third, cycle K and all (k) -cycles are in their cycle order in the supercycle. Fourth, the interiors of (k) -cycles are not examined in extending K to a supercycle. Fifth, some (k) -cycles are separated into fragments by the supercycle; see, for example, Figure 3.18 (the supercycle consists of the dashed lines), where each (k) -cycle (A and A') is fragmented into two paths.

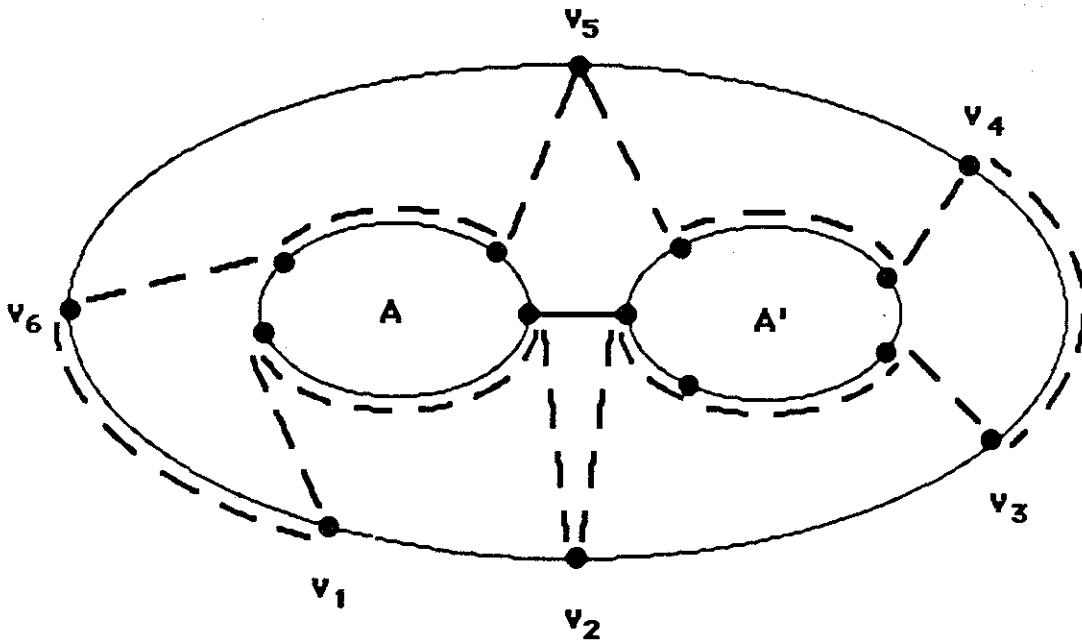


Figure 3.18. Supercycle Fragments (k) -Cycles

The problem with this method is in the fifth feature. The vertices of a (k) -cycle may be mingled among level $k-1$ vertices. This mingling cannot be prevented at successive levels. Thus, there may be crossings of edges separated by an arbitrary number of levels, so there is no way to reuse pages from one level at any later level and obtain a bound on the number of pages required. The solution to this problem is discussed next. Unfortunately, as the solution eliminates the problem of the fifth feature, it also eliminates the advantage of the second feature.

3.3.3. Nesting

We wish to prevent the crossing of edges from levels that are far apart. We want to accomplish this by preventing the mingling of vertices from levels $\leq k-1$ with the vertices of a (k) -cycle. We can do so if the supercycle extending cycle K to include $G_k|K$ contains each (k) -cycle contiguously. This is a requirement in addition to the requirement of maintaining each D-cycle in cycle order in the supercycle.

The key to placing all the vertices of a D-cycle contiguously in the supercycle is that once one vertex of the cycle is reached, all the vertices must be picked up. In general, this cannot be accomplished without adding edges to G . Since the interior of G is maximal planar, these new edges destroy planarity. To restore planarity, a few carefully selected edges are deleted, and other edges are re-routed. This process of adding, deleting and re-routing edges to obtain a supercycle is called *micro-surgery*.

The subgraph of interest is composed of K , $B_{k-1,k}|K$ and $G_k|K$. The exterior of K and the interiors of the (k) -cycles are of no current interest, so we proceed as though they were not present in the planar embedding. Micro-surgery is used when a vertex u_j of some (k) -cycle A is added to the supercycle between v_q and v_{q+1} of cycle K : all remaining vertices of A must immediately be added to the supercycle. Suppose the edge (u_j, u_{j+1}) is on A , and both u_j and u_{j+1} are in P_{v_q} . Suppose further that u_j is the rightmost vertex of P_{v_q} that is on A . Micro-surgery starts by deleting edge (u_j, u_{j+1}) , which creates a gap between u_j and u_{j+1} in the planar embedding. These edges $(v_q, u_{j+2}), (v_q, u_{j+3}), \dots, (v_q, u_s)$ are re-routed through the gap. The result is that these edges are

now attached to the vertices of cycle A from the inside rather than the outside; see Figure 3.19. The re-routing opens space to add the edge (u_{j+1}, u_{s+1}) if u_{s+1} exists, or the edge (u_{j+1}, v_{t+1}) otherwise. The supercycle is then extended around A (in a clockwise direction in Figure 3.19) from u_j to u_{j+1} and thence to u_{s+1} or v_{t+1} .

What can go wrong? While cycle A is being traversed from u_j to u_{j+1} , a vertex of another cycle might be added to the supercycle. An example is shown in Figure 3.20 where the pinch vertex z is shared by both cycle A and cycle A' . This presents a dilemma. The vertices of A' cannot be picked up at this point because we must keep the vertices of A contiguous on the supercycle. However, if the vertices of A' are not picked up at this point, then vertex z will not be contiguous with the remaining vertices of A' on the supercycle. Therefore, we are forced to relax the require-

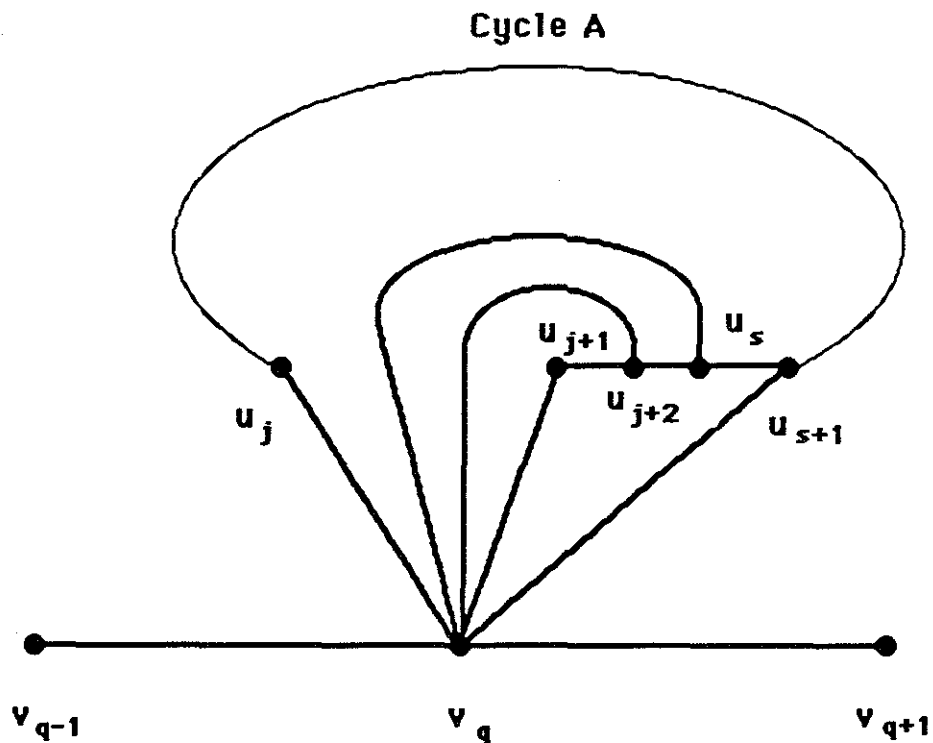


Figure 3.19. Micro-Surgery on a (k) -Cycle

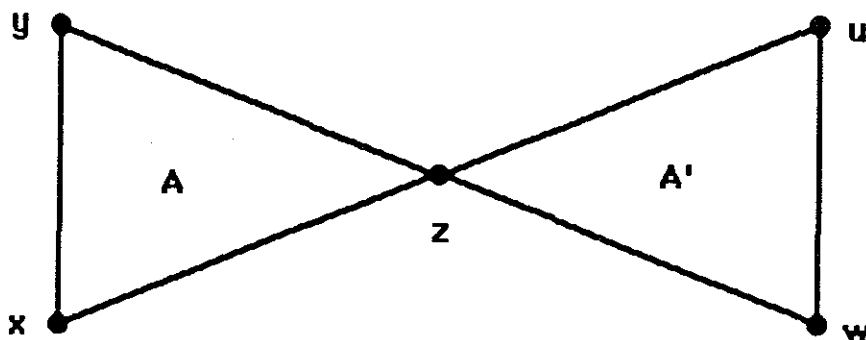


Figure 3.20. Vertex z Shared by Two (k) -Cycles

ment, because of these pinch vertices.

The requirement on the supercycle that we will be able to meet is:

Each (k) -cycle appears contiguously in the supercycle, except for at most one distinguished vertex.

In Figure 3.20, the distinguished vertex for (k) -cycle A' is z . Since z is not contiguous with the remaining vertices of A' , z is said to be *separated* from A' .

We must guarantee that at most one vertex is separated from each cycle. We justify the requirement with the BC-tree T of $G_k|K$; see Figure 3.21. Recall that T is obtained from $G_k|K$ by performing the following operation on each cycle A of $G_k|K$: add a vertex v_A in the interior of A , add an edge from v_A to each vertex of A , and remove all edges of A . The process of extending K to a supercycle can be thought of as a traversal of T . The traversal is depth-first until the first cycle-edge is encountered. Suppose that cycle-edge is on A . At that point, the vertex v_A pre-empts all vertices of A for itself (all vertices of A are immediately picked up). This can be viewed as a contraction of all vertices of A to v_A ; see Figure 3.22. In particular v_A pre-empts z from cycle A' . When later the edge (z,w) of cycle A' is encountered, all vertices of cycle A' except z are pre-empted by vertex $v_{A'}$. See Figure 3.23. It is clear that in a depth-first traversal of T , vertices z

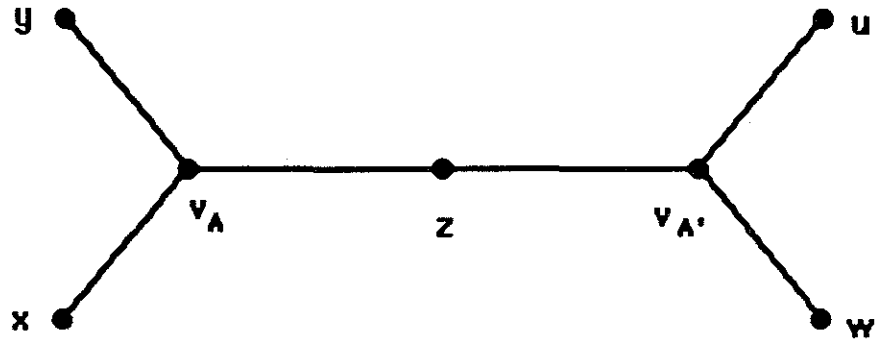


Figure 3.21. The BC-Tree

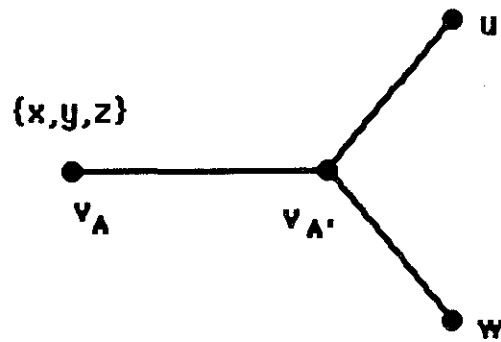


Figure 3.22. Contraction of Cycle A to v_A

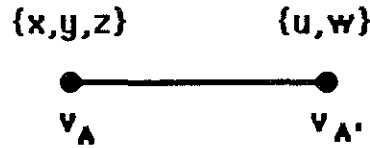


Figure 3.23. Contraction of Cycle A' Except z to v_A

and w will be encountered before any other vertices of A' and that vertex w will be reached through the vertex $v_{A'}$. Thus z is the only vertex separated from A' .

We have established the following micro-surgery lemma.

Lemma 3.11. Using micro-surgery, a $(k-1)$ -cycle K can be extended to a supercycle S containing $G_k|K$ such that

- (1) each (k) -cycle of $G_k|K$ is in cycle order in S ;
- (2) each (k) -cycle of $G_k|K$ is contiguous in S except for at most one separated vertex.

We now have the supercycle we want. At successive levels, the supercycle nests each (k) -cycle between two consecutive vertices of a $(k-1)$ -cycle. For each (k) -cycle, we have to account for three categories of pages: the pages for the edges from its father $(k-1)$ -cycle to the (k) -cycle; the pages for the edges from the (k) -cycle to its son nodes; and the pages from the (k) -cycle to its separated vertex and from the (k) -cycle to its brother (k) -cycles for their separated vertices that are on the (k) -cycle. We show that the number of pages required in each category is bounded at every level. Because of nesting, the pages used at one level can be reused at subsequent levels. Thus, nesting provides the last element of the solution to a bounded pagenumber for planar graphs.

3.4. Levels Without Cycles

In this section, we address in detail the case where some level does not contain any cycles. It is sufficient to consider the case where G_1 contains no cycles. The approach is to extend the cycle G_0 to a superhamiltonian cycle including G_1 . In fact, we can show that G is actually hamiltonian in this case. No edges must be added to G to obtain the desired superhamiltonian cycle. As promised in the last section, the problems with chordal edges are resolved by an algorithm presented in this section.

Lemma 3.12. If G_1 contains no cycles, then G_1 is a forest, and every level greater than 1 is empty.

Proof: If G_1 contains no cycles, then G_1 is a forest by definition. No level greater than 1 can be nonempty since there are no level 1 cycles to contain level 2 vertices. \square

Lemma 3.13. If G_1 contains no cycles and is connected, then G_1 is a tree.

Proof: By the definition of a tree. \square

Theorem 3.14. If G_1 contains no cycles and if X_0 is empty, then G_1 is connected; moreover, there exists a hamiltonian cycle H for G such that the vertices of G_0 appear in H in the same order as they do in the cycle G_0 .

Proof: We construct H using micro-surgery; the construction is simplified by the absence of (1)-cycles in G . By Lemma 3.5, G_1 is connected. By Lemma 3.13, G_1 is a tree. Let v_1, \dots, v_m be the vertices of G_0 in cyclic (say clockwise) order. Since X_0 is empty, all interior vertices of the path $P_{v_q}, 1 \leq q \leq m$, are in V_1 . First, suppose that G_1 is empty. Since G is inner-triangulated, and since X_0 is empty, G is a triangle, and the theorem is easily satisfied. Now assume that G_1 is nonempty. Then for each v_q , there exists some vertex of G_1 adjacent to v_q ; in other words, each P_{v_q} has length greater than one.

We state the construction of H by micro-surgery for the case in which G_1 contains no (1)-cycles. Start H at v_1 . Whenever $v_q, 1 \leq q \leq m$, is reached, examine P_{v_q} for vertices not yet in H . Let $(v_{q-1}, u_1, \dots, u_b, v_{q+1})$ denote P_{v_q} (subscripts of the v 's are taken modulo m). If each

$u_j, 1 \leq j \leq k$, is already in H , then extend H from v_q to v_{q+1} via the edge (v_q, v_{q+1}) ; if $v_{q+1} = v_1$, then halt. Otherwise, let j be smallest such that u_j is not in H . Extend H from v_q to v_{q+1} via the path $(v_q, u_j, u_{j+1}, \dots, u_k, v_{q+1})$; if $v_{q+1} = v_1$, then halt. Figure 3.17 shows in dashed lines the extension of H from v_q to v_{q+1} . It is clear that every vertex of G is in H .

It remains to show that no vertex of G_1 is visited twice during the construction of H . For the purposes of obtaining a contradiction, suppose that $u_r \in V_1$ is visited twice, the second time while H is being extended from v_q to v_{q+1} . Consider Figure 3.24. Since u_r is not yet in H , we know that $u_j \neq u_r$; however, u_{j+1} and u_r may be identical. Since (u_j, u_{j+1}) is a non-cycle edge, by Lemma 3.7, there exists $v_s \in V_0$ such that $v_s \neq v_q$, and edge (u_j, u_{j+1}) is visible from v_s . Let v_t be the vertex of G_0 at which u_r was first visited. Since v_t is visited before v_q , and since G_0 is visited in clockwise order, we have $t \leq s < q$. Therefore, v_s is visited before v_q ; in particular, u_j is visited before v_q . But this is a contradiction to u_j not having been visited before. This contradiction

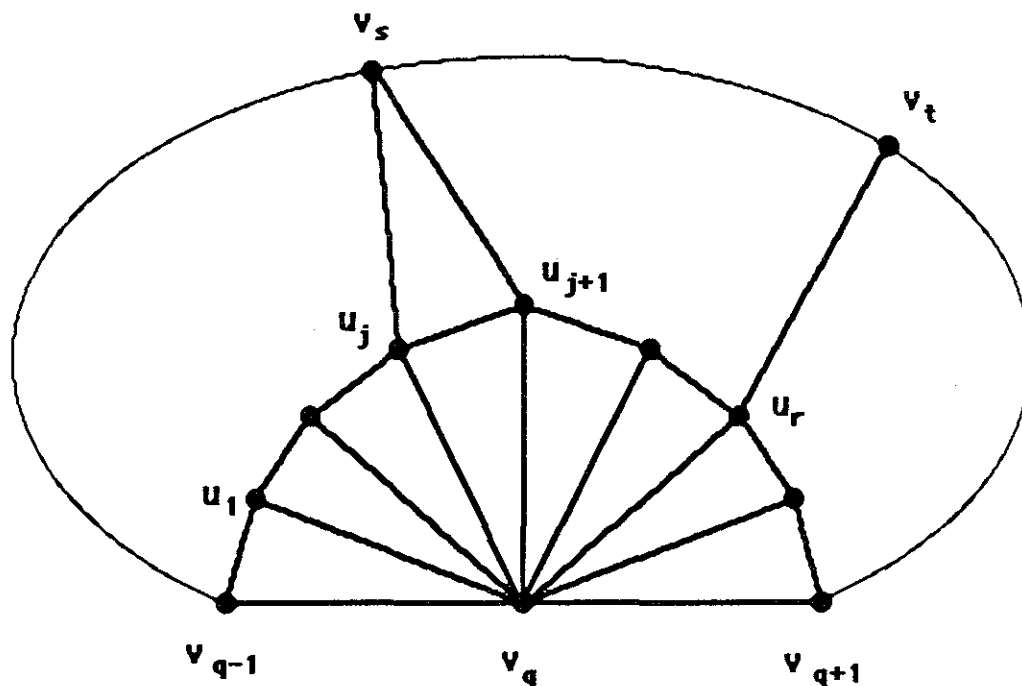


Figure 3.24. Contradiction for Theorem 3.14

validates the construction of H .

Clearly H contains V_0 in the order v_1, \dots, v_n . Hence H is the desired hamiltonian cycle for G . \square

Theorem 3.15. If G_1 contains no cycles, then G is subhamiltonian; moreover, the order of the vertices of G_0 in the hamiltonian cycle is preserved.

Proof: Assume G_1 is not empty. If X_0 is empty, apply Theorem 3.14. If X_0 is nonempty, re-embed the X_0 edges outside G_0 (this can be done by outerplanarity), and triangulate the interior of G_0 in such a way that no new chordal edges are introduced. This triangulation is always possible for the following reason. With the X_0 edges removed, any interior face of G must have a level 1 vertex on its boundary. Thus each interior face can be triangulated by connecting level 1 vertices to level 0 vertices or to other level 1 vertices in such a way that no cycles are created in G_1 . Now, Theorem 3.14 applies to G_0 and its interior. Edges of X_0 will necessarily be exterior to the hamiltonian cycle in the modified embedding. \square

We apply Theorem 3.15 to the graph G of Figure 3.12. G_1 contains no cycles and is not connected. Hence, we re-embed the edges of $X_0 = \{(v_2, v_4), (v_4, v_{10}), (v_5, v_9), (v_7, v_9)\}$ outside G_0 and retriangulate the interior of G_0 . Figure 3.25 shows the result. The curved edges are those from X_0 . Edges (v_3, u_3) , (u_2, u_5) , (u_4, u_6) and (v_8, u_6) have been added so that the interior of G_0 is now retriangulated, G_1 is connected, and G_1 has no cycles (hence is a tree). We apply the algorithm of Theorem 3.14 to obtain a hamiltonian cycle, which is shown by dashed edges in Figure 3.26.

Applying Proposition 2.5 and Theorem 3.15, we have the following:

Corollary 3.16. If $C_1 = \emptyset$, then G is two-page embeddable.

3.5. The Algorithm

In this section, the development of our algorithm for embedding a planar graph in a seven-page book culminates in a description and analysis of the algorithm. The components that have been obtained in previous sections are combined into an integrated unit. The reasons for the result seven are given. The pagewidth of the resulting embedding is discussed. The linear time

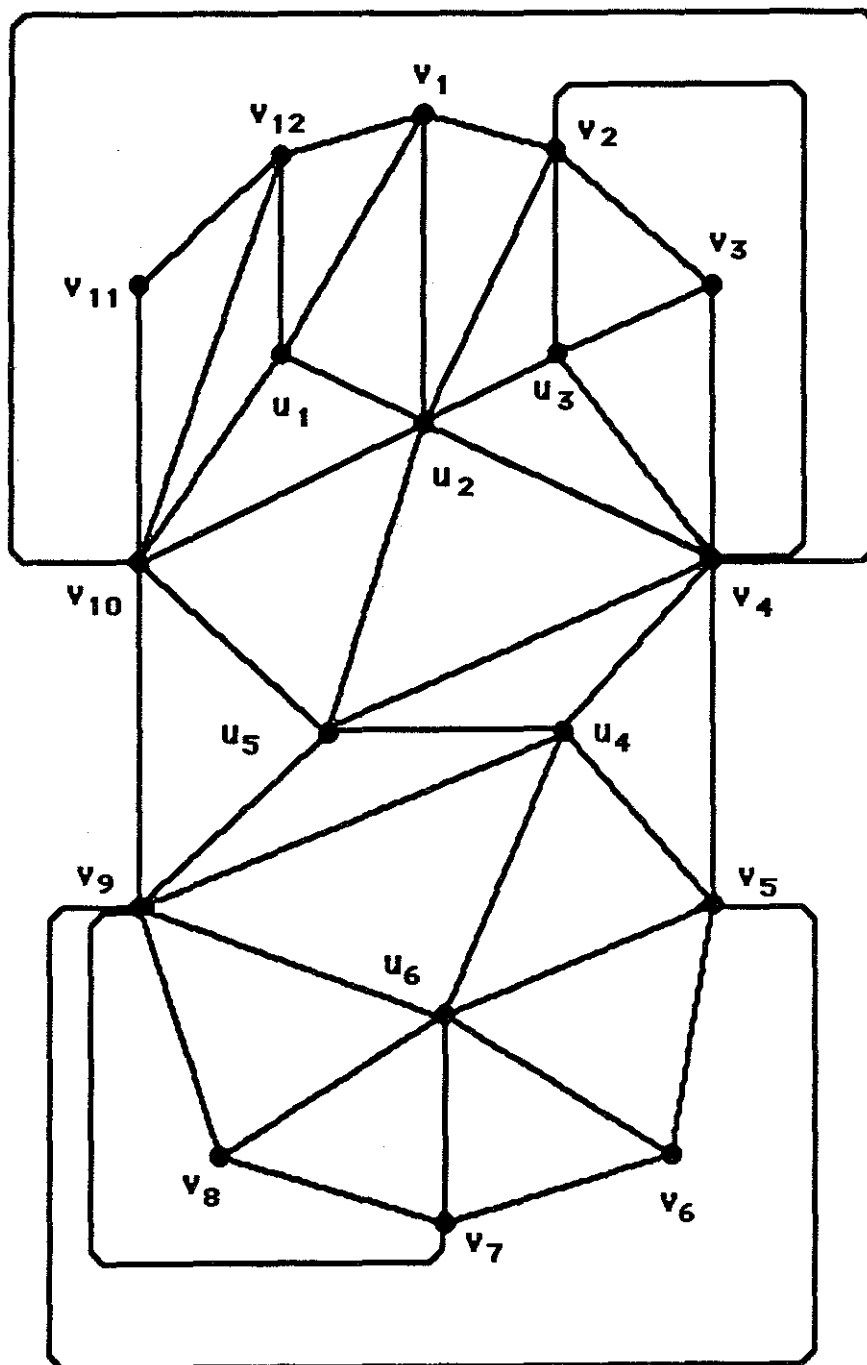


Figure 3.25. Retriangulated Graph

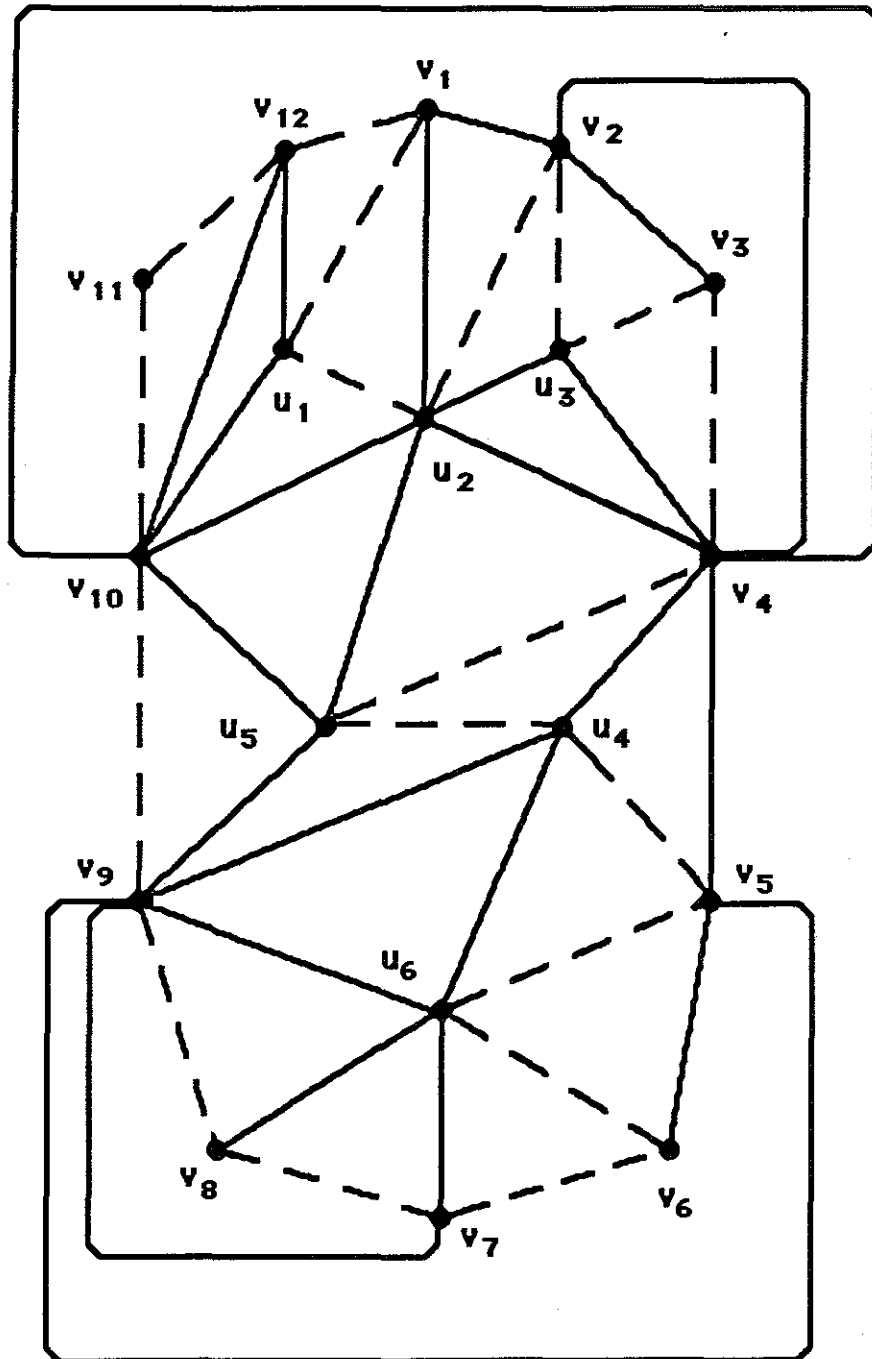


Figure 3.26. Hamiltonian Cycle

performance is verified.

3.5.1. The Statement

Before stating the algorithm, we need two definitions. If K is a (k) -cycle, let T be the BC-tree of $G_{k+1} | K$. Suppose we have chosen a root for T . Define the distance from the root to v_A by the number of cycles between the root and v_A . Call A *even* if the distance to v_A is even, and call A *odd* otherwise. The significance of this terminology is that two odd cycles cannot share a pinch vertex. We have to account only for pinch vertices between odd/even cycle pairs.

We start with a condensed statement of the algorithm. Given an I-graph G , visit the levels of G in a breadth-first manner (by traversing its D-tree). Each (k) -cycle K of G is assigned a distinguished vertex $first(K)$ and two pages $page(K)$ and $avoid(K)$ by the algorithm. K is extended to a supercycle including $G_{k+1} | K$ by micro-surgery. The problem of chordal edges is taken care of as in the previous section. The supercycle gives the embedding of the vertices of $G_{k+1} | K$ with respect to the embedding of K . The set of pages is the set of natural numbers $\{1, 2, \dots\}$. In the algorithm, each of a , b , and c is a variable taking on a page as value. The edges incident to $first(K)$ are assigned to $page(K)$, as are all other edges outside the supercycle. The edges inside the supercycle are assigned to a second page c . The BC-tree of $G_{k+1} | K$ is constructed. The even cycles of $G_{k+1} | K$ are assigned to a page a and the odd cycles to a page b . The pages a , b and c must be other than page $avoid(K)$. Each cycle K is assigned a $first(K)$ value; if K has a separated vertex, then $first(K)$ is that vertex.

A complete statement of the steps of the algorithm is given in Algorithm 3.1. Each step will be discussed in turn in the remainder of this subsection. The choice of pages a , b and c in steps 4.4 and 4.5 is nondeterministic. While the choice could be made deterministically, the non-determinism effectively indicates the freedom available in the algorithm.

(1) The D-tree T of G is constructed by breadth-first search of G from G_0 , the root of T . This search identifies the levels and the (k) -cycles of G . Since the search can be accomplished in $O(|E|)$ time, and since G is planar, this step requires $O(|V|)$ time. In practice, this step would be

- (1) Construct the D-tree of G by breadth-first search.
- (2) Choose an arbitrary vertex of G_0 to be $first(G_0)$.
- (3) Embed G_0 in a circle in its cycle order. Assign page 1 to be $page(G_0)$.
- (4) For each (k) -cycle K , visit its sons by executing steps 4.1 through 4.5.
- (4.1) Embed the chordal edges $X_k|K$ outside K , and retriangulate the interior of K so that no new chordal edges are introduced.
- (4.2) By micro-surgery, extend K to a supercycle S including all vertices of $G_{k+1}|K$.
- (4.3) Nest the embedding of $G_{k+1}|K$ within the embedding for K in the order given by S .
- (4.4) Choose two pages a and b that are not incident to any vertex of K . For each $(k+1)$ -cycle A in $G_{k+1}|K$, set $page(A)=a$ and $avoid(A)=b$ if A is even and set $page(A)=b$ and $avoid(A)=a$ if A is odd.
- (4.5) Let c be a page different from a and b and from any page incident to vertices of K . Assign the exterior edges of S to $page(K)$ and the interior edges to c .

Algorithm 3.1. Planar Graph Algorithm

accomplished at the same time as the remaining steps.

(2) This is an initialization step. Every (k) -cycle K of G will be assigned a vertex $first(K)$ that is meant to be its separated vertex, if any. It is also the vertex at which the supercycle extending K is to begin. Since G_0 will have no separated vertex, an arbitrary vertex is chosen to be $first(G_0)$. Since $first(K)$ is the start of the supercycle, all edges incident to $first(K)$ are either on or outside the supercycle. Thus all edges incident to $first(K)$ can be assigned to the same page. That page will be $page(K)$. For G_0 , we arbitrarily choose $page(G_0)=1$.

(3) With this step, the book embedding actually begins. The vertices of G_0 are placed on a circle in cycle order. The vertex $first(G_0)$ is the first vertex in the embedding.

(4) Here is the breadth-first traversal of the D-tree. A visited (k) -cycle K is extended by visiting all of its sons during the execution of steps 4.1 through 4.5. Breadth-first is not the only traversal scheme that will work, but it will do. The important properties of the traversal are that a father be visited before its sons and that all brothers be visited "simultaneously." At the point that K is extended, we know that all vertices of K have been embedded and are contiguous in that embedding, except perhaps for $first(K)$. None of the edges on K or inside K have been

assigned to pages, but the edges of K will be assigned to pages in this execution of step 4.5.

(4.1) The chordal edges in K may prevent $G_{k+1}|K$ from being connected. $G_{k+1}|K$ can be made connected by the operations used in Theorem 3.15. Cross edges are pulled outside K , and the interior of K is retriangulated to make $G_{k+1}|K$ connected. Note that all edges of $X_k|K$ will be on the outside of the supercycle obtained in step 4.2. Therefore, these edges will all be assigned to $page(K)$ in step 4.5. The solution of the problem of chordal edges thus comes with no additional page cost.

(4.2) $G_{k+1}|K$ is modified by the micro-surgery technique of section 3.3 to allow K to be extended to a supercycle S including $G_{k+1}|K$. Let A be any cycle of $G_{k+1}|K$. Then the vertices of A are in cycle order in S . In addition, the vertices of A are contiguous in S except perhaps for a separated vertex u , which is assigned to be $first(A)$.

(4.3) The supercycle S gives the embedding order for the vertices of $G_{k+1}|K$ with respect to the vertices of K . The vertices of K are currently contiguous in the embedding, except possibly for $first(K)$. Suppose the vertices of K are $first(K) = v_1, v_2, \dots, v_n$. Then the picture before $G_{k+1}|K$ is embedded is as in Figure 3.27, where v_2, \dots, v_n are contiguous in the embedding, but there may be other vertices between v_1 and v_2 . The creation of S causes all vertices of P_{v_1} to be

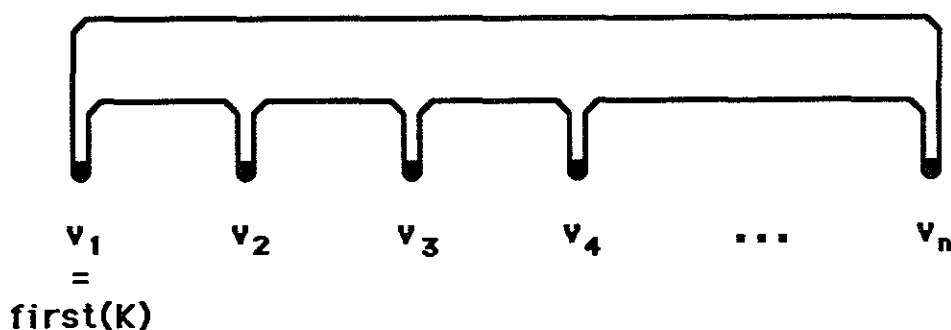


Figure 3.27. K Before $G_{k+1}|K$ Is Embedded

in S before v_2 . There will be other vertices before v_2 if any $(k+1)$ -cycle edges are encountered in P_{v_1} . Denote by S_{v_1} all the vertices between v_1 and v_2 in the supercycle. All other vertices of $G_{k+1} \setminus K$ occur in S before v_n . Thus, all of S_{v_1} can be placed next to v_2 , and the remainder of S will fit in the interval between v_2 and v_n . The result is as in Figure 3.28 where the vertices of $S \setminus K$ are shown larger than v_1, \dots, v_n .

(4.4) Let A be any $(k+1)$ -cycle in $G_{k+1} \setminus K$ and let $u_1 (= \text{first}(A)), u_2, \dots, u_m$ be the vertices of A . The edges of A will not be assigned to pages until its sons are visited. (However, the non-cycle edges $N_{k+1} \setminus K$ will be assigned to pages in this execution of step 4.5.) From step 4.3, it is clear that no edges of A can cross any other edges already embedded, except for edges incident to u_1 . The edges incident to u_1 consist of (u_1, u_2) and (u_1, u_m) , as well as edges from u_1 to $G_{k+2} \setminus K$. Call this set of edges the *separated edges* of u_1 . Any $u_i \neq u_1$ might be separated from some cycle other than A . For example, see Figure 3.28 where u_2 is separated from cycle A' and u_3 is separated from cycle A'' . At least the separated edge (u_1, u_m) will cross the separated edges of u_2 and u_3 , but no separated edge of u_2 crosses a separated edge of u_3 . Thus A must be assigned a

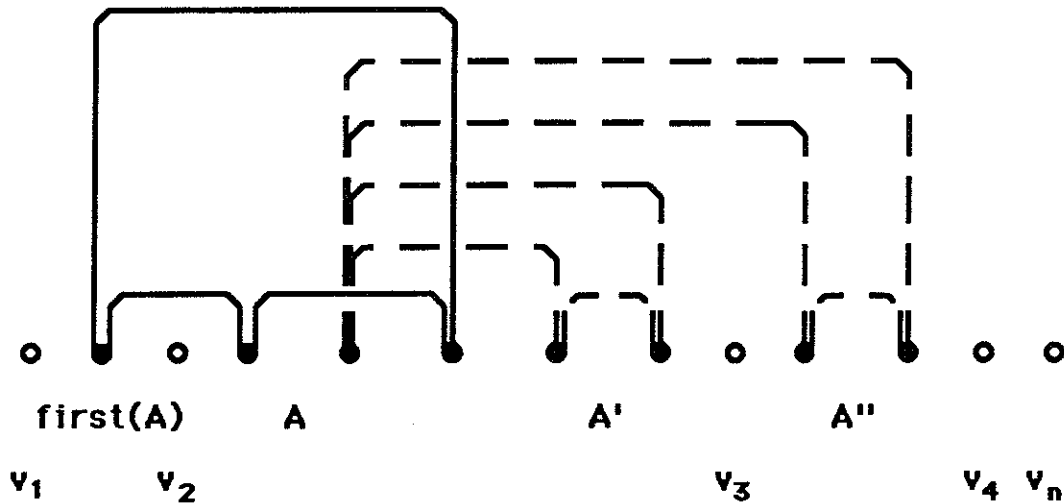


Figure 3.28. The Embedding of S

page different from the page of both A' and A'' , but A' and A'' may be assigned the same page. This conflict between separated edges potentially occurs at any cycle of $G_{k+1} | K$. Translating to the language of the BC-tree of $G_{k+1} | K$, the even $(k+1)$ -cycles may all be assigned the same page a while the odd $(k+1)$ -cycles may all be assigned the same page b , but a must be different from b . We choose a and b to be different from any page incident to K and from $page(K)$. Then setting $page(A)=a$ if A is even and $page(A)=b$ if A is odd avoids the page conflict.

(4.5) This is the page assignment step. The edges on K , the chordal edge $X_t | K$, the binding edges $B_{k,k+1} | K$ and the non-cycle edges $N_{k+1} | K$ are assigned to one of two pages according to whether they are inside or outside of S . (Edges on S may be assigned to either page, except the edges incident to $first(K)$ must be assigned to $page(K)$.) The edges outside of S are assigned to $page(K)$. Since all edges incident to $first(K)$ are either outside S or on S , all edges incident to $first(K)$ are assigned to $page(K)$. The interior edges of S are assigned to page c . Page c is chosen to differ from all other pages incident to K , including $page(K)$ and to differ from pages a and b of step 4.4.

This completes the description of the algorithm. The verification of the correctness of the algorithm comes from the following theorem. The bounded pagenumber is proved later.

Theorem 3.17. Algorithm 3.1 yields a valid book embedding of an I-graph G .

Proof: Most of the proof has already been accomplished. Certainly Algorithm 3.1 embeds the vertices of G on a circle. It remains only to show that the page assignments in step 4.5 never introduce two crossing edges assigned to the same page. But this is clear: $page(K)$ was chosen specifically so that the separated edges of $first(K)$ could safely reach K . The vertices of the super-cycle S other than $first(K)$ are contiguous in the embedding. The only additional edges that can cross edges of S are those incident to K from the outside, i.e., those embedded when the parent of K was expanded. Hence the choices of pages a , b and c cannot raise a conflict.

Thus the algorithm yields a book embedding for G . \square

3.5.2. Why Seven?

So far we have a valid book embedding for an I-graph G . To obtain a bounded pagenumber, we constrain our algorithm to select its pages from a bounded set. This is possible because of the independence of alternate levels caused by the nesting of cycles. The analysis of the exact number of pages in that set begins with the following result.

Lemma 3.18. Let K be any (k) -cycle of G . Book embed G using Algorithm 3.1. Then at most five pages are incident to K . (A page is *incident* to K if there is an edge incident to a vertex of K that is assigned to that page.)

Proof: Suppose $K=G_0$. Then at the first execution of step 4.5, all edges incident to G_0 are assigned to pages $page(G_0)$ and c . Hence at most two pages are incident to G_0 .

Suppose $K \neq G_0$. Let K' be the parent of K in the D-tree. Then edges incident to K that are exterior to K divide into two classes. The first class comprises those edges assigned to pages when step 4.5 is executed for K' . There are two pages used there. The second class comprises the edges separated from cycles of $G_t|K'$ other than K . These are all assigned to page $avoid(K)$. Hence, three pages suffice for all incident edges exterior to K . The edges on K or interior to K are assigned to one of two pages when step 4.5 is executed for K . Hence five pages suffice for all edges incident to K . \square

Theorem 3.19. Algorithm 3.1 can choose from a set of seven pages. As a result, any planar graph can be embedded in a book of seven pages.

Proof: Consider any (k) -cycle K and the execution of Algorithm 3.1 at the point that K is expanded to a supercycle. After step 4.5 is executed, all edges incident to K will have been assigned to pages. By Lemma 3.18, at most five pages are incident to K . The choices for pages a and b are constrained only to avoid conflict with those five pages. Hence a set of seven pages suffices. \square

3.5.3. Further Analysis

We consider the pagewidth of the book embedding of G . If G does not have bounded degree, then there is no hope of bounding the pagewidth in a bounded page embedding. For example, if G is a *star*, it has a vertex of degree $\Omega(|V|)$. Then any bounded page embedding of G requires $\Omega(|V|)$ pagewidth.

So assume G has bounded degree. Even here our algorithm can give poor pagewidth. We define a sequence of planar graphs called the cylinder of triangles. The k th cylinder of triangles CT_k consists of vertices $V = \{a_j, b_j, c_j | 1 \leq j \leq k\}$ and edges

$$E = \{(a_j, b_j), (a_j, c_j), (b_j, c_j) | 1 \leq j \leq k\} \cup \{(a_j, a_{j+1}), (b_j, b_{j+1}), (c_j, c_{j+1}) | 1 \leq j < k\}.$$

A drawing of CT_3 is shown in Figure 3.29. The leveling of CT_k is obvious; each of the k levels consists of a single triangle. Algorithm 3.1 will completely nest these triangles, giving a $\Theta(|V|)$ pagewidth in a four page embedding. Any two-page embedding of CT_k has pagewidth $\Omega(|V|)$; however, there exists an $O(1)$ pagewidth embedding of CT_k in three pages [CLR]. Thus there is much room to improve the pagewidth performance of Algorithm 3.1.

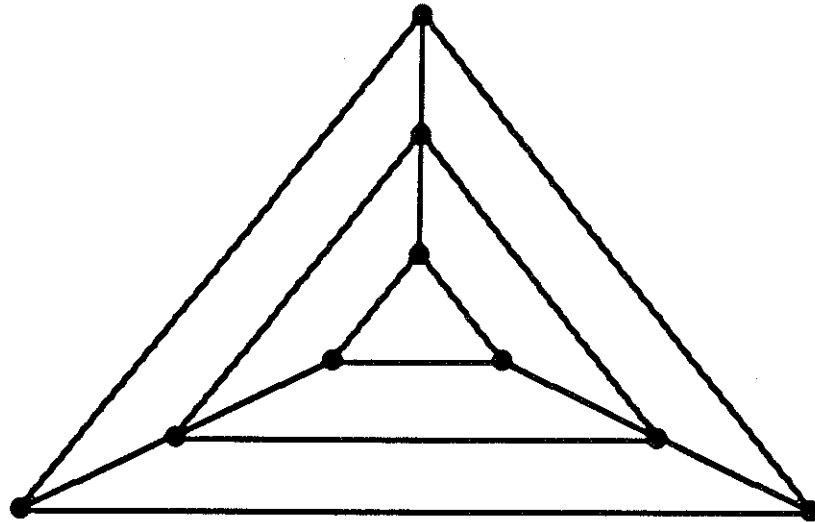


Figure 3.29. A Cylinder of Three Triangles

It is obvious that Algorithm 3.1 executes in linear time. The breadth-first search is certainly linear. Each vertex and edge is visited at most twice, and all elementary operations on vertices and edges can be done in constant time. Theorem 3.19 can thus be strengthened to:

Theorem 3.20. Any planar graph can be embedded in a seven-page book in linear time.

3.6. Conclusions

In this chapter, we have made progress towards determining the pagenumber of the class of planar graphs. The best bounds currently known are $3 \leq PPG \leq 7$. The gap between these bounds presents two challenges.

The first challenge is to raise the lower bound. The argument that $PPG \geq 3$ is relatively simple. If indeed PPG is greater than 3, there is a need for improved lower bound techniques. Our approach towards such techniques is discussed in the concluding chapter.

The second challenge is to lower the upper bound. We are the first to attain the seven page upper bound, and we do so with a time-optimal algorithm. While it is possible to show upper bounds on pagenumber nonconstructively [CLR], we do not believe the structure of planar graphs is amenable to a nonconstructive proof of small upper bounds. Therefore, we require an new algorithm to lower the upper bound. Our algorithm has some slack in its page assignments, but it is not clear how to exploit the slack to obtain smaller pagenumber. Our algorithm is targeted to the book embedding problem for planar graphs, so the principles on which it is based should not be ignored in a search for other algorithms. A thoughtful modification of these principles might suffice to obtain an improved upper bound. In particular, we feel there may be a better approach to choosing the leveling than ours or Buss and Shor's.

Three other types of problems can be raised. First, we can seek an algorithm that gives a bounded pagenumber for a larger class of graphs. For example, we have tried to extend Algorithm 3.1 to the class of genus one graphs but were unable to do so. Second, we can seek an algorithm for a subclass of planar graphs that gives a pagenumber less than seven. This is the approach we take in the next chapter where we show that the class of trivalent planar graphs has

pagenumber two. Third, we can seek to improve the pagewidth performance of our algorithm. In chapter 4, we do so for bounded-degree outerplanar graphs.

CHAPTER 4

EMBEDDING TRIVALENT PLANAR GRAPHS IN TWO PAGES

In this chapter, we consider the pagenumber of planar graphs of restricted valence. In particular, we seek bounds on the valence of a planar graph that guarantees that the graph is subhamiltonian (two-page embeddable). Our main result is that all trivalent planar graphs are subhamiltonian. Here we take *trivalent* to mean that every vertex has degree no greater than three, but we do not require regularity. From the proof of this result, we obtain an algorithm for obtaining a two-page embedding for a trivalent planar graph. Our algorithm executes in time linear in the size of the graph. Along the way, we develop a method of traversing the faces of a biconnected planar graph.

4.1. Overview of the Algorithm

Let G be a trivalent planar graph. Throughout we assume that a particular planar embedding of G is given. To show that G is two-page embeddable, we construct a superhamiltonian cycle for G . Our approach is to add edges to some of the faces of the planar embedding of G and to demonstrate a hamiltonian cycle in the resulting graph.

The following result of Bernhart and Kainen [BK] allows us to reduce the general case to the case where G is biconnected.

Proposition 4.1 [BK] For any graph G , the pagenumber of G equals the maximum of the pagenumbers of its biconnected components.

Therefore, if each biconnected component of G is subhamiltonian, then G is subhamiltonian. Thus, it is sufficient to consider only biconnected G .

The notion of adjacency places an important role in our results. A face of G is *adjacent* to the edges and vertices of G on its boundary. Adjacency is symmetric; e.g., if a face is adjacent to an edge, then the edge is adjacent to the face. Two faces of G are *adjacent* if there is an edge of G adjacent to both.

If G is biconnected, each face of a planar embedding is bounded by a cycle. Our proof that a biconnected trivalent planar G is subhamiltonian is by induction on the number of faces of G . The face added in the inductive step is always adjacent to the exterior face. The added face is formed by choosing two vertices on the exterior face and appending a new path with those vertices as endpoints. Since the graph is trivalent and biconnected, these two vertices must have degree two before the face is added.

As illustration of the idea of an added face, see Figures 4.1 and 4.2. In Figure 4.1, G is a biconnected trivalent planar graph with vertices x and y of degree two on its exterior face. In Figure 4.2, G' has been constructed by appending path (z_1, z_2, z_3) to x and y , creating the face F . G' is also a biconnected trivalent planar graph, and it has one face more than G . To extend a superhamiltonian cycle H for G to a superhamiltonian cycle H' for G' , we must assume that the edges (x, u) and (v, y) are in H . Figure 4.3 shows the modification of H to obtain H' . We replace path (v, y, t) in H by edge (v, t) in H' ; we replace edge (x, u) in H by path (x, z_1, z_2, z_3, y, u) in H' . For

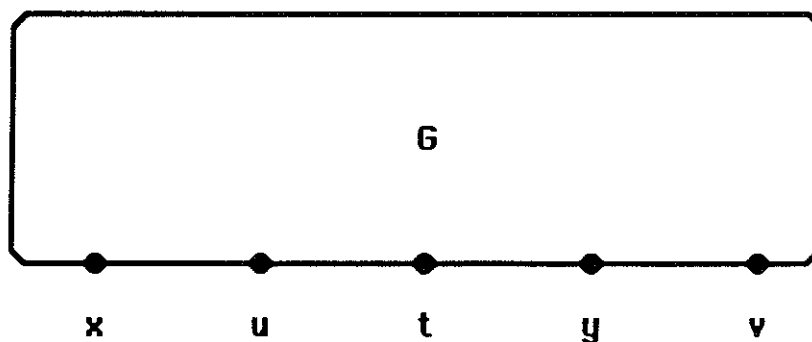


Figure 4.1. Biconnected Trivalent Planar Graph

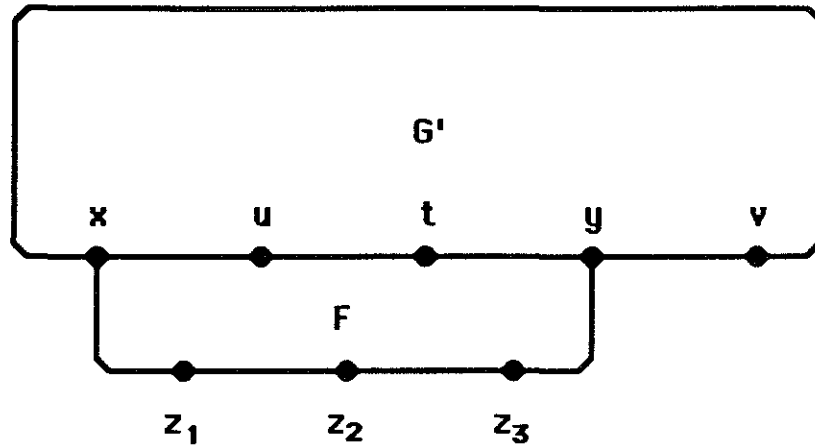


Figure 4.2. Creation of Face F

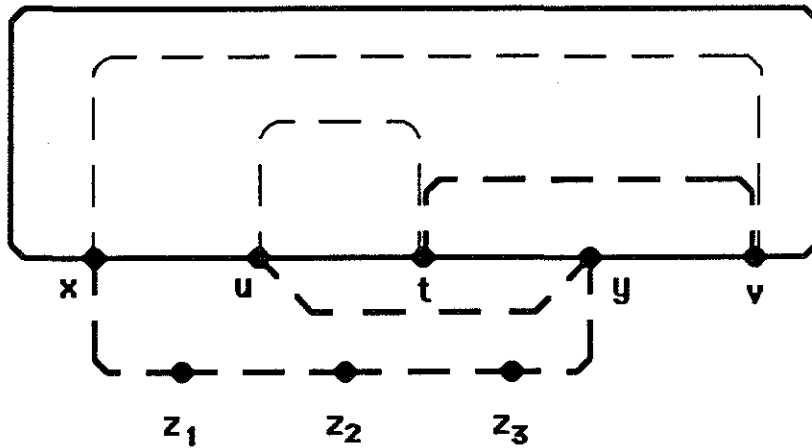


Figure 4.3. Superhamiltonian Cycle H'

this to succeed, we require the following: for any degree two vertex u on the exterior face of G , the edge of G incident to u in a counterclockwise direction is in the superhamiltonian cycle H' . This requirement guarantees that edge (y, v) is in H , and justifies the replacement of the path

(t,y,v) by edge (t,v) .

The structure of biconnected planar graphs is examined in the next section. The resulting structural information is applied to the proof of the main theorem in the following section. From the main theorem, we derive an algorithm for embedding trivalent planar graphs in two pages in Section 4.4. The algorithm assumes that a biconnected planar graph is constructed by adding one face at a time in a suitable order. This order, called *oriented face traversal*, is developed in Section 4.5.

4.2. Structure of Biconnected Planar Graphs

In this section, we are primarily interested in the structure of faces adjacent to the exterior face. If G is a biconnected planar graph with a given planar embedding, let U_G (or simply U , if G is clear from context) be the unbounded (or exterior) face of G . Call a face F of G a *boundary face* if F is adjacent to U . For a boundary face F , let $G_F = (V_F, E_F)$ be the vertices and edges of the bounding cycle of F that are adjacent to U . That is, G_F is the intersection of the bounding cycles of U and F .

The boundary faces of G can be partitioned into three classes:

- I. G_F is a path or G_F is the entire bounding cycle of F . In the second case, the only faces of G are U and F , and $G = G_F$. See the example in Figure 4.4. G_F consists of the path (v_1, v_2, v_3, v_4) .
- II. G_F is a path together with one or more isolated vertices. See the example in Figure 4.5. G_F consists of the path (u_1, u_2, u_3) together with isolated vertices s and t .
- III. G_F is two or more disjoint paths and zero or more isolated vertices. See the example in Figure 4.6. G_F consists of the two paths (v_1, v_2) and (u_1, u_2, u_3) together with isolated vertex s .

Since G_F is a subgraph of a cycle and contains at least one edge, these three classes exhaust the possibilities.

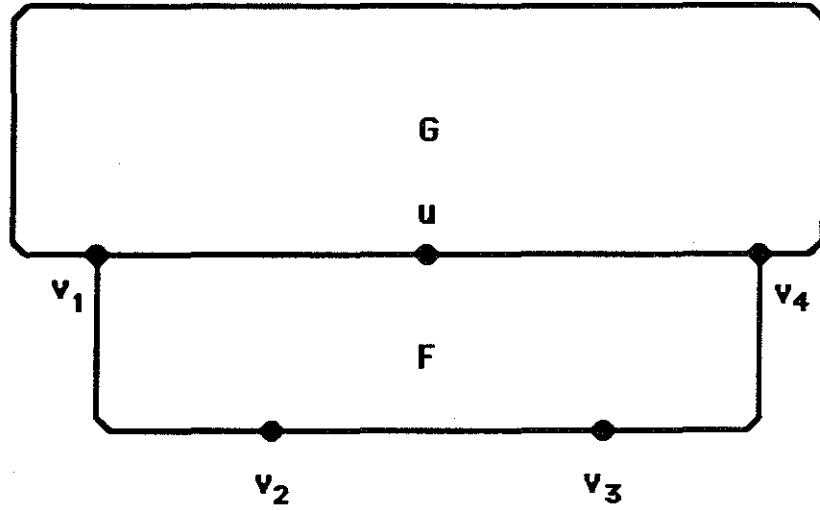


Figure 4.4. Class I Face

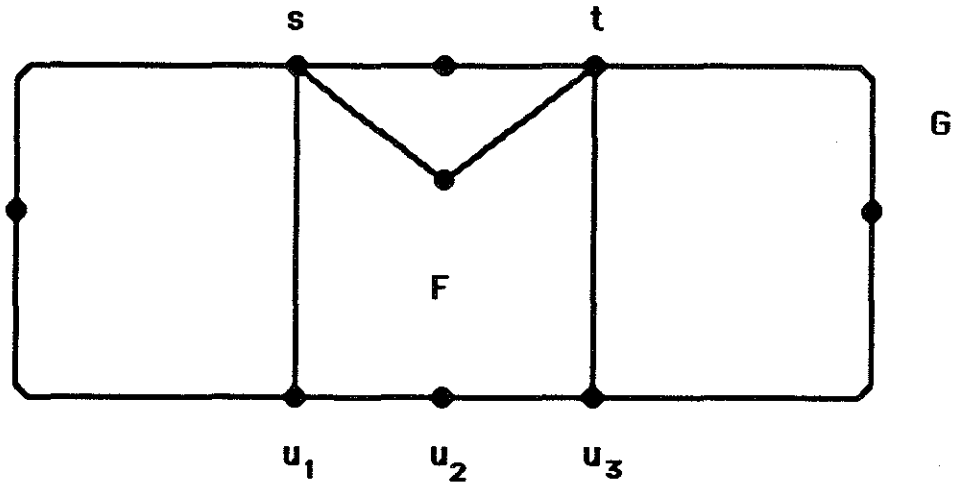


Figure 4.5. Class II Face

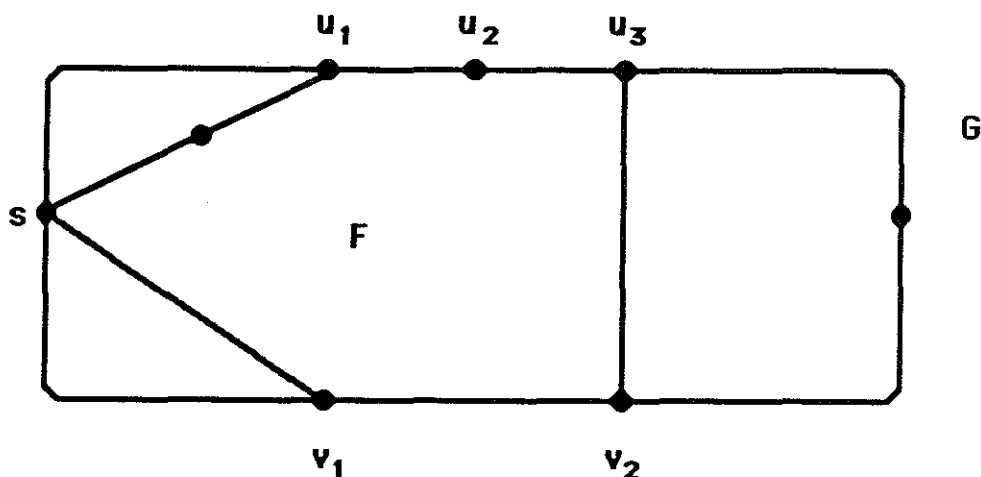


Figure 4.6. Class III Face

Lemma 4.2. Let F be a boundary face of a biconnected trivalent planar graph G . Then G_F is either of Class I or of Class III without isolated vertices.

Proof: It suffices to show that G_F contains no isolated vertices. To obtain a contradiction, suppose s were an isolated vertex of G_F . Since s is adjacent to U , two edges incident to s are adjacent to U . Similarly, two edges incident to s are adjacent to F . The degree of s is either two or three. Hence some edge e incident to s is adjacent to both U and F . But then e is in G_F , and s is not isolated. This contradiction proves the lemma. \square

It is clear from the proof that we can construct a biconnected quadrivalent planar graph G that has a boundary face F such that G_F contains isolated vertices. Hence, the Lemma cannot be extended to valences greater than three.

This emphasis on the adjacency of faces suggests that we consider the concept of the dual of a planar graph (Even [Ev]). Let $G=(V,E)$ be a planar graph with a given planar embedding. The dual of G is a multigraph $G^D=(V^D,E^D)$ where V^D is the set of faces of G , and E^D contains one edge for each edge of E : if $e \in E$ is adjacent to faces $F, F' \in V^D$, then $e^D=(F, F')$ is an edge in E^D , and these are the only elements of E^D . Note that G^D can contain parallel edges and loops. If

G is biconnected, G^D does not contain loops. Parallel edges in G^D cause no problems in our context (see the discussion of Algorithm 4.3). It is clear that G^D is also planar; we always use the obvious planar embedding of G^D that is derived from the one for G .

An example of a dual graph is given in Figure 4.7. Here the edges of G are dashed while the edges of G^D are solid. G has three Class I faces, one Class III face and one non-boundary face. There are seven edges incident to U in G^D , only three of which are completed in the figure. Note that there are two parallel edges between the Class III face and the Class I face on the right.

We are particularly interested in the boundary faces of G . Let $BF(G)=(V^B, E^B)$ (or simply BF), the *boundary face graph* of G , be the subgraph of G^D induced by the boundary faces of G . For trivalent graphs, there is a simple condition that guarantees the adjacency of two faces in BF .

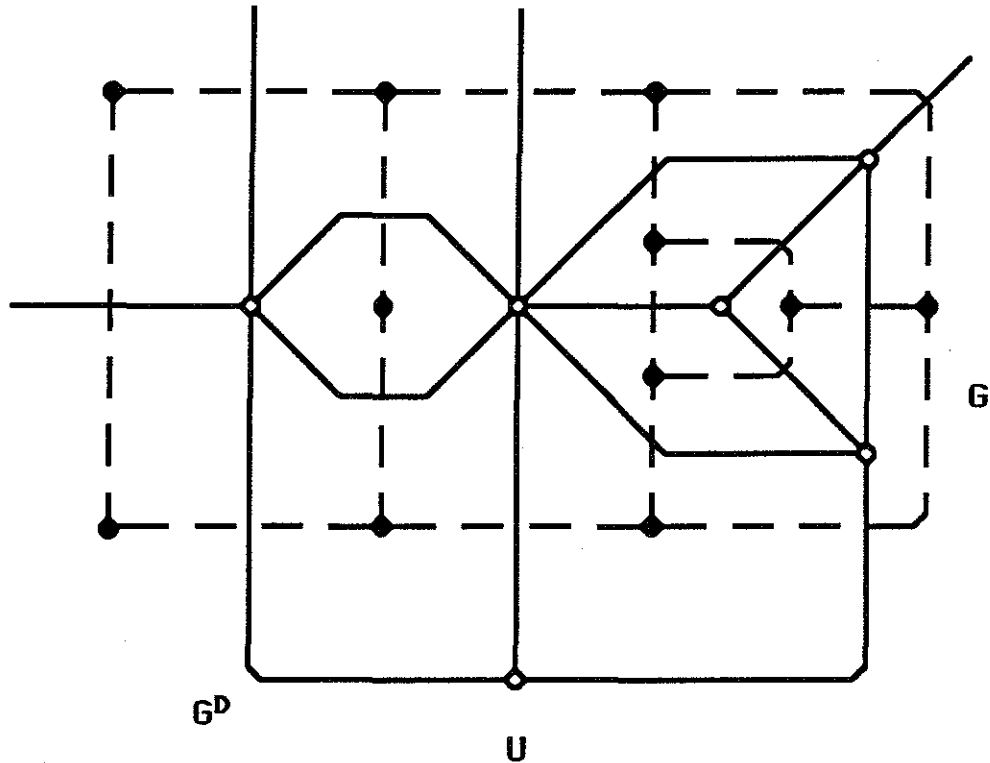


Figure 4.7. Dual Graph G^D

Lemma 4.3. Let G be a biconnected trivalent planar graph. Let (v_1, v_2, v_3) be any path on the bounding cycle of U . Let F be the boundary face adjacent to (v_1, v_2) and let F' be the boundary face adjacent to (v_2, v_3) . If v_2 is of degree two, then $F = F'$. If v_2 is of degree three, then F and F' are adjacent in BF .

Proof: If v_2 is of degree two, then there is no edge to separate F from F' . So suppose that v_2 is of degree three, that u is different from v_1 , and that v_3 is adjacent to v_2 . Then the edge (v_2, u) is adjacent to both face F and face F' . Hence F and F' are adjacent in BF . \square

We have the following result on the structure of BF .

Lemma 4.4. Let G be a biconnected trivalent planar graph. Then BF is a connected outerplanar graph.

Proof: Since each face in V^B is adjacent to U , BF is clearly outerplanar. To show that BF is connected, let $F, F' \in V^B$ be distinct faces and let $P = (v_1, v_2, \dots, v_j)$ be a subpath of the bounding cycle of U such that (v_1, v_2) is on the bounding cycle of F and (v_{j-1}, v_j) is on the bounding cycle of F' . We say that P joins F and F' . Clearly $j \geq 3$. We show that there is a path from F to F' in BF . We proceed by induction on j .

If $j=3$, then since F and F' are distinct, Lemma 4.3 implies that F and F' are adjacent in BF . Now suppose that $j > 3$ and that the result holds for smaller values than j . Let $F'' \in V^B$ be the boundary face such that (v_{j-2}, v_{j-1}) is on the bounding cycle of F'' . If $F'' = F'$, then $(v_1, v_2, \dots, v_{j-1})$ is a shorter path joining F and F' ; by induction, there is a path in BF from F to F' . If $F'' = F$, then (v_{j-2}, v_{j-1}, v_j) is a path joining F and F' ; by Lemma 3.3, F and F' are adjacent in BF . Otherwise, by Lemma 4.3, F'' and F' are adjacent in BF . Since $(v_1, v_2, \dots, v_{j-1})$ is a path joining F and F'' that is shorter than j , by induction there is a path in BF between F and F'' . But now there is a path from F to F' in BF . This completes the induction. Therefore, BF is connected. \square

The following lemma characterizes the Class I faces of G . Recall that a *cutpoint* of a graph is a vertex whose removal disconnects the graph.

Lemma 4.5. Let G be a biconnected trivalent planar graph. Then $F \in V^B$ is a Class I face if and only if F is not a cutpoint of BF .

Proof: Suppose F is not a Class I face. By Lemma 4.2, F is a class III face without isolated vertices. Let P_1 and P_2 be two distinct maximal paths of G_F . Choose edges e_1 in P_1 and e_2 in P_2 . Draw a closed curve from F through e_1 to U and back to F through e_2 . This closed curve can be taken to consist of the edges $e_1^D, e_2^D \in E^D$. The result is illustrated in Figure 4.8. There are faces of G distinct from F and U both inside and outside this curve. In particular, there are elements of $V^B - \{F\}$ both inside and outside this curve. Any path in BF from the inside to the outside must cross this curve and hence must pass through F . Hence F is a cutpoint of BF .

Now suppose F is a cutpoint of BF . Let F' and F'' be in different components of $BF - \{F\}$. Then F' and F'' are not adjacent in BF . Choose $e' \in G_{F'}$ and $e'' \in G_{F''}$. Starting at e' , traverse the

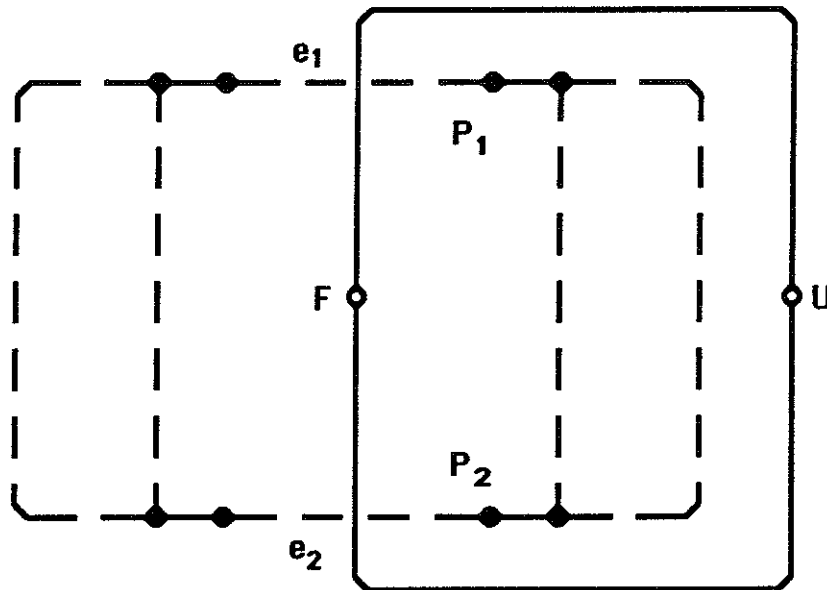


Figure 4.8. Curve Through a Class III Face

bounding cycle of U in the clockwise direction to reach e'' . Name the faces encountered F_1, F_2, \dots, F_k in order. By Lemma 4.3, F_j is adjacent to F_{j-1} for $1 \leq j < k$. Since F' and F'' are in different components of $BF - \{F\}$, $F = F_j$ for some $1 < j < k$. Furthermore, there is some maximal path P_{cw} in G_F that is encountered in the clockwise traversal from e' to e'' . If a similar traversal from e' to e'' is taken in the counterclockwise direction, then a maximal path P_{ccw} in G_F is encountered. But P_{cw} and P_{ccw} are disjoint. Hence F is in Class III, i.e., not in Class I. \square

Corollary 4.6. Let G be a biconnected trivalent planar graph. Then V^B contains at least one face of Class I.

Proof: In any (nonempty) connected graph, there exists some vertex that is not a cutpoint. To see this, use a depth-first search of the graph to obtain a rooted spanning tree. From Lemma 4.6 of [Ev], we deduce that the leaves of this tree are not cutpoints of the graph. Since BF is connected (Lemma 4.4), it contains a vertex that is not a cutpoint. By Lemma 4.5, that vertex is a Class I face of G . \square

The inductive step of our main theorem is based on two graphs one of which is obtained by adding a single face of Class I to the other. Let G be a biconnected planar graph, and let F be a Class I face of G . Then G_F is a path P on the bounding cycle of the exterior face U . Define $G' = G - F$ to be G without the edges and interior vertices of P . (If $G_F = G$, then $G' = G - F$ is the empty graph.) Figure 4.9 shows the result of subtracting face F from the graph of Figure 4.4.

Lemma 4.7 plays a crucial role in the inductive step of the main theorem of this chapter. It also provides part of the correctness proof for the book embedding algorithm of Section 4.4.

Lemma 4.7. Let G be a biconnected planar graph and let F be a Class I face of G . Then $G' = G - F$ is a biconnected planar graph.

Proof: It is clear that G' is planar. It remains to show that G' is biconnected. If G' is empty, then it is trivially biconnected. So assume that G' is not empty. Let $P = (u_1, u_2, \dots, u_j)$ be the maximal path in G_F . Let P' be the path from u_1 to u_j in the bounding cycle of F that contains no edges of P . Let P'' be the path in the bounding cycle of U_G from u_1 to u_j that contains no edges of P . Figure 4.10 shows the situation described. By the definition of Class I, no edge and

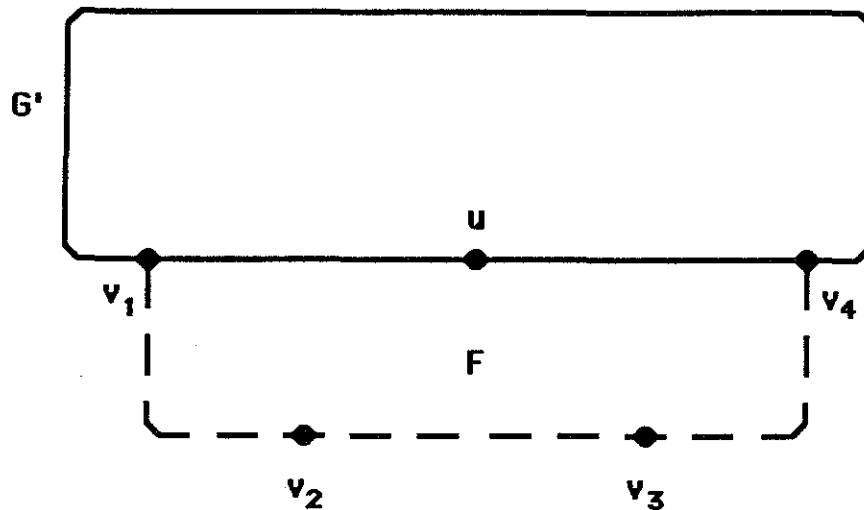


Figure 4.9. $G' = G - F$

no interior vertex of P' is adjacent to U_G . Thus P' and P'' constitute the boundary of $U_{G'}$, and this boundary is a cycle. All other faces of G' are the same as faces of G . Hence every face of G' is bounded by a cycle, and G' is biconnected. \square

4.3. The Main Theorem

Using the knowledge gained about the structure of biconnected trivalent planar graphs, we can prove the main result of this chapter.

Theorem 4.8. Let G be a biconnected trivalent planar graph. Then G is subhamiltonian. Furthermore, a superhamiltonian cycle H for G can be constructed with these two characteristics:

- (i) each edge in $H - E$ is embedded in an interior face of G ;
- (ii) if (v_1, v_2, v_3) is a path on the bounding cycle of U in counterclockwise order and v_2 has degree two, then the edge (v_2, v_3) occurs in H .

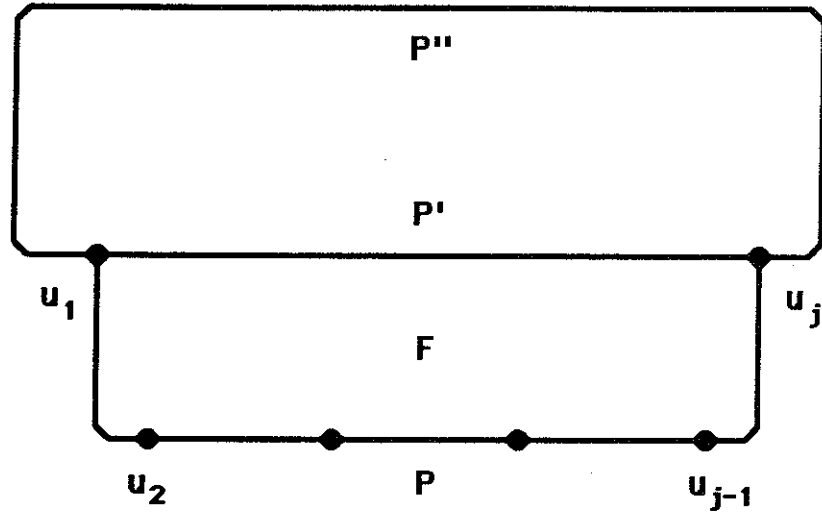


Figure 4.10. Proof of Lemma 4.7

Proof: We proceed by induction on the number of faces of G . The statement of the inductive hypothesis follows.

If G is a biconnected trivalent planar graph and has $m \geq 1$ interior faces, then G has a superhamiltonian cycle H satisfying characteristics (i) and (ii).

If G has one interior face, then G is a cycle, and $H = G$ trivially satisfies characteristics (i) and (ii).

For purposes of induction, assume that G has $m > 1$ interior faces and that we have shown the truth of the inductive hypothesis for $m-1$ interior faces. By Corollary 4.6, G has a Class I face F . Let $G' = G - F$. By Lemma 4.7, G' is a biconnected planar graph. Since G' is a subgraph of G , it is also trivalent. G' has $m-1$ interior faces. By inductive hypothesis, G' has a superhamiltonian cycle H' satisfying characteristics (i) and (ii).

Let $P=(u_1, u_2, \dots, u_j)$, $j \geq 2$, be the maximal path in G_F in a counterclockwise direction. Let $P'=(u_1, v_1, \dots, v_k, u_j)$ be the path on the bounding cycle of F that avoids G_F . Note that P' may consist of only the edge (u_1, u_j) , in which case $i=0$. Let $s \neq u_{j-1}$ be adjacent to u_j on the bounding cycle of U . Note that if $j > 2$ and $i > 0$, we may have $s = u_1$. The situation so far is illustrated in Figure 4.11.

It is clear that u_1 and u_j have degree three in G and hence have degree two in G' . By characteristic (ii), H' includes either edge (u_1, v_1) if $i > 0$ or edge (u_1, u_j) if $i = 0$ as well as edge (u_j, s) . We have three cases to consider.

Case 1. $j=2$. Here P is just the edge (u_1, u_2) and the vertex sets of G and G' are identical. Thus H' is a superhamiltonian cycle for G satisfying characteristics (i) and (ii).

Case 2. $j > 2$ and $i = 0$. Here P' is just the edge (u_1, u_j) , and H' includes that edge. Let H be the cycle gotten from H' by deleting the edge (u_1, u_2) and adding the path P . Then H is a superhamiltonian cycle for G' satisfying characteristics (i) and (ii).

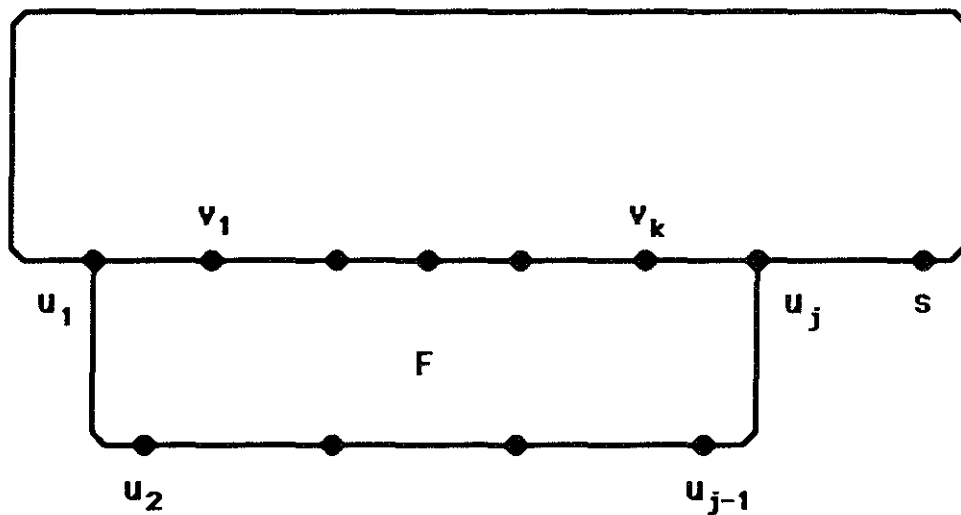


Figure 4.11. Proof of Theorem 4.8

Case 3. $j > 2$ and $i > 0$. Both P and P' are paths of length at least two. By characteristic (ii), H' contains edges (u_1, v_1) and (u_j, s) . Figure 4.12 illustrates the crucial edges in H' . H is obtained from H' in two steps.

In the first step, u_j is eliminated from H' . Let F' be the interior face of G' adjacent to u_j . Since u_j has degree two in G' , both v_k and s are adjacent to F' . Let $t \neq s$ be adjacent to u_j in H' . By characteristic (i), either $t = v_k$, or the edge (t, u_j) is embedded in F' . In either case, the edge (t, u_j) can be removed from H' and the edge (t, s) can be embedded in F' to obtain a supercycle H'' . The result remains planar and H'' is a supercycle of G' containing every vertex of $G' - \{u_j\}$. The result of this first step is shown in Figure 4.13.

In the second step, the edge (u_1, v_1) in H'' is replaced by the path $(u_1, u_2, \dots, u_j, v_1)$. Note that (u_j, v_1) is an added edge that is embedded in F . Let H be the resulting supercycle. The result of this second step is shown in Figure 4.14. H is clearly a superhamiltonian cycle for G .

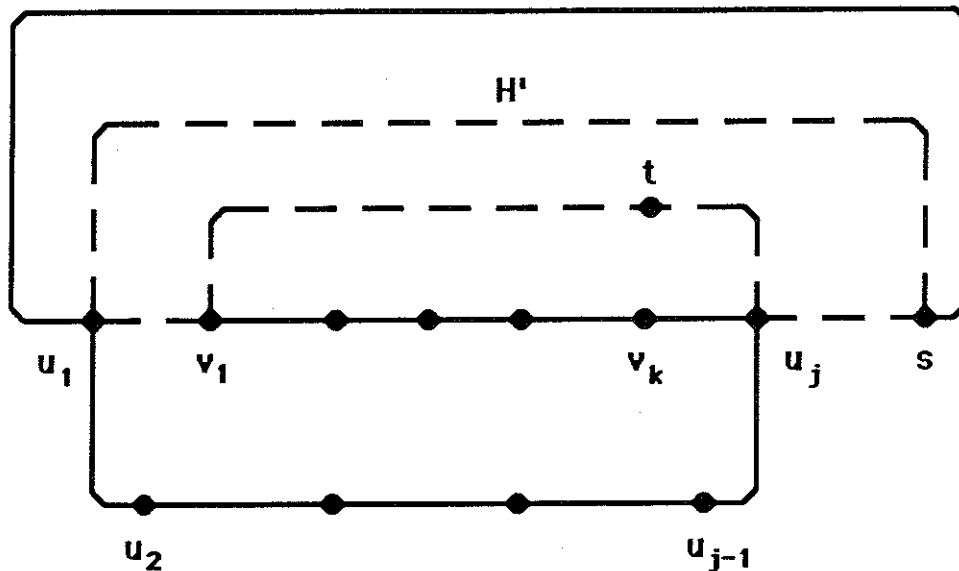


Figure 4.12. The Superhamiltonian Cycle H'

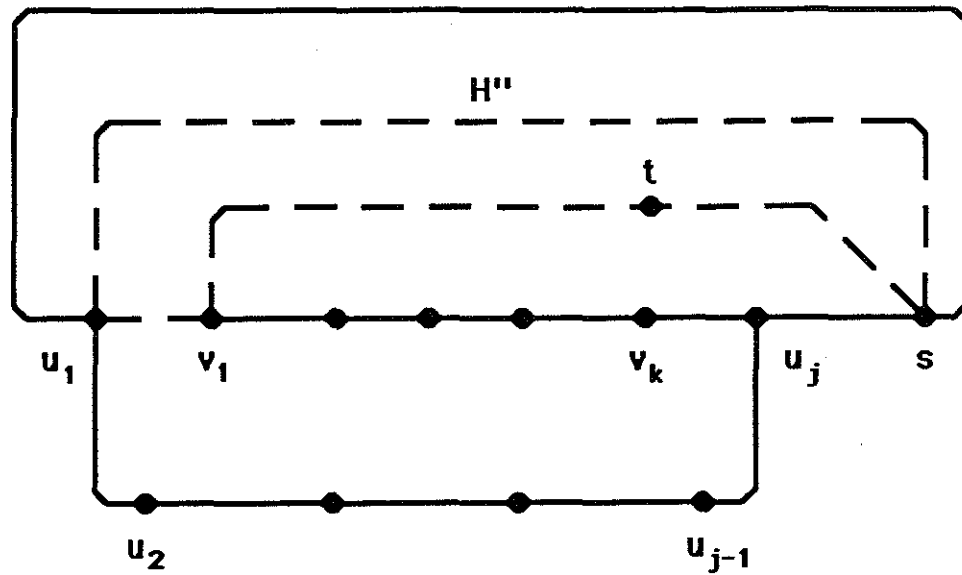


Figure 4.13. The Supercycle H''

H satisfies characteristic (i) since H' satisfies it and since the only edges added are (u, v_1) and (t, s) which are embedded in interior faces of G .

To show that characteristic (ii) holds for H , suppose that x, y, z is a path in counterclockwise order on the bounding cycle of U_G , and suppose that y has degree two in G . Since u_1 and u_j have degree three in G , $y \neq u_1$ and $y \neq u_j$. Consider the possibilities. If $\{x, y, z\} \cap \{u_1, u_2, \dots, u_j\} = \emptyset$, then $\{y, z\}$ occurs in H by inductive hypothesis. If $z = u_1$, then there are two subcases. The first subcase is $y = u_j$, which we have already eliminated. The second subcase is $y \neq u_j$; then $\{y, z\}$ occurs in H by inductive hypothesis. If $z \neq u_1$ and $z = u_j$, we have $y = s$. The first step constructing H'' did not eliminate (s, z) which occurs in H' and hence in H by inductive hypothesis. Finally, if $y = u_r, 1 < r < j$, then edge $(u_r, u_{r+1}) = (y, z)$ is in H by the second step constructing H from H'' . Under all possibilities, H satisfies characteristic (ii).

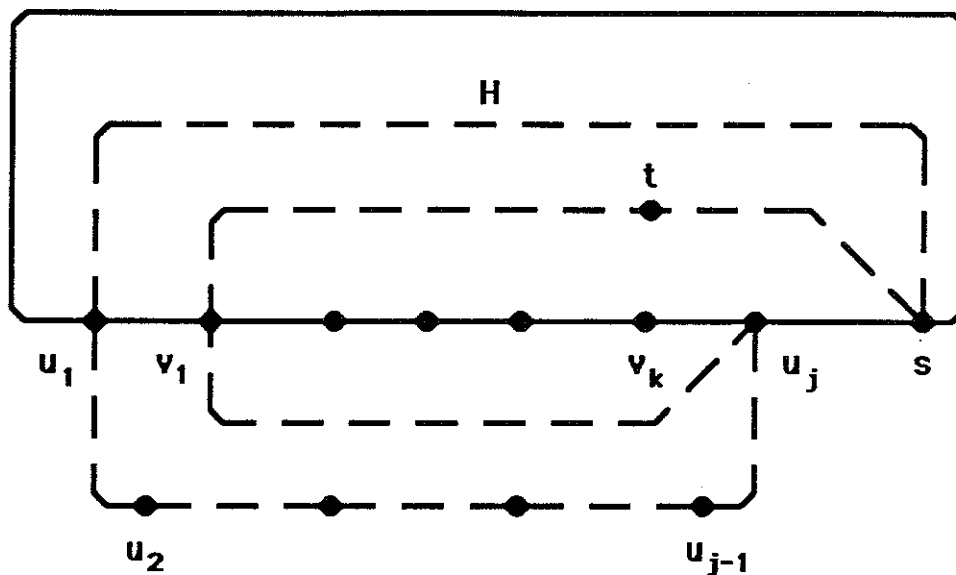


Figure 4.14. The Superhamiltonian Cycle H

This completes the proof for Case 3.

By induction, the theorem follows. \square

We restate our results in terms of book embeddings.

Corollary 4.9. Every trivalent planar graph is two-page embeddable.

Proof: By the remarks of section 4.1, and by Theorem 4.8. \square

A construction for a superhamiltonian cycle of a biconnected trivalent planar graph is implicit in the proof of Theorem 4.8. In the next section, we develop an explicit algorithm for embedding a trivalent planar graph in two pages in time linear in the size of the graph.

4.4. The Algorithm

This section presents our algorithm for embedding a trivalent planar graph in a two-page book. Let $G=(V,E)$ be a trivalent planar graph. We apply Algorithm 4.1 to G to obtain a two-

- (1) Use depth-first search to find the biconnected components of G .
- (2) Embed each biconnected component in a two-page book with Algorithm 4.3.
- (3) Combine the two-page embeddings into a single two-page embedding of G .

Algorithm 4.1. Two-Page Embedding of a General Trivalent Planar Graph

page embedding. Algorithm 4.1 reduces the general case to the case of a biconnected trivalent planar graph in step 2. That case is handled in Algorithm 4.3 (described later in this section). The correctness of Algorithm 4.1 is self-evident, assuming the correctness of Algorithm 4.3.

We analyze the time complexity of Algorithm 4.1 based on the assumption (to be proven later) that Algorithm 4.3 operates in time linear in the size of its input. To show that Algorithm 4.1 executes in linear time, we generalize it to an arbitrary k -page embeddable (not necessarily planar) graph G in Algorithm 4.2.

Lemma 4.10. Let G be a graph with pagewidth $\leq k$. If there exists an algorithm for embedding each biconnected component C of G in a k -page book in time linear in $|C|$, then Algorithm 4.2 embeds G in a k -page book in time linear in $|G|$.

Proof: The determination of the biconnected components of G by depth-first search in step 1 executes in time $O(|E|)$ [Ev]. Each edge is in exactly one biconnected component of G . Consider

- (1) Use depth-first search to find the biconnected components of G .
- (2) Embed each biconnected component in a k -page book.
- (3) Combine the k -page embeddings of the biconnected components into a single k -page embedding of G .

Algorithm 4.2. Embedding a Graph with Pagewidth at Most k

all the biconnected components input to step 2. If G has no isolated vertices, then the size of that input is $O(|E|)$. Even if G has isolated vertices, the size of that input is $O(|G|)$. Hence, step 2 executes in time linear in $|G|$.

We assume that circular linked lists represent the vertex order of the k -page embeddings. Let v be a cutpoint of G and let C_1 and C_2 be two biconnected components of G having v in common. We claim that the k -page embeddings for C_1 and C_2 can be combined into a k -page embedding for $C_1 \cup C_2$ in constant time. Let $\cdots uvw \cdots$ be the vertex order for C_1 , and let $\cdots zvy \cdots$ be the vertex order for C_2 . The vertex order for $C_1 \cup C_2$ is $\cdots uvy \cdots zw \cdots$, which can be obtained by breaking the two circular linked lists at v and creating a single circular linked list. There are at most $|G|$ combinations to perform in step 3, each executing in constant time. Hence Algorithm 4.2 executes in time linear in $|G|$. \square

Corollary 4.11. Algorithm 4.1 executes in time $O(|G|) = O(|V|)$.

The problem reduces to embedding a biconnected trivalent planar graph G in a two-page book in linear time. Following the cue of the inductive proof of Theorem 4.8, we wish to construct a finite sequence of biconnected trivalent planar graphs terminating in G such that successive members of the sequence are generated by the addition of Class I faces. We also say that $G = G' + F$ when F is constructed by attachment of a path to two vertices on the bounding cycle of the exterior face of G' . Define an *addition sequence* for G to be a sequence of biconnected trivalent planar graphs G_1, G_2, \dots, G_m such that

- (1) G_1 is a cycle;
- (2) $G_k = G$;
- (3) $G_k = G_{k-1} + F_k$, $1 < k \leq m$;
- (4) F_k is a Class I face of G_k .

Call each F_k an *addition face*. Define $G_m, G_{m-1}, \dots, G_2, G_1$ to be a *subtraction sequence* for G if $G_1, G_2, \dots, G_{m-1}, G_m$ is an addition sequence for G . Call m the *length* of the sequence.

Lemma 4.12. Let G be a biconnected trivalent planar graph with m interior faces. Then there exists a length m addition sequence for G .

Proof: Using Corollary 4.6 and Lemma 4.7, it is an easy induction on m to show that there exists a subtraction sequence for G . But then there exists an addition sequence for G . \square

We now introduce Algorithm 4.3 for constructing a superhamiltonian cycle of a biconnected trivalent planar graph. The correctness of the algorithm is immediate, as it is merely a restatement of the inductive proof of Theorem 4.1. It remains to show that it executes in linear time. The proof that an addition sequence can be constructed in linear time (step 1) is deferred to the next section. Clearly constant time suffices for step 2. Since G is trivalent, it is easy to see that each vertex and each edge of G is visited $O(1)$ times during all executions of steps 4.1 and 4.2. The work done in each execution of steps 4.1 and 4.2 is proportional to the size of F_k . The sum of the sizes of the F_k 's is linear in $|V|$. The net result is that Algorithm 4.3 executes in $O(|V|)$ time. We summarize the results of this section in the following theorem.

Theorem 4.13. Let $G=(V,E)$ be a trivalent planar graph. Then G can be embedded in a two-page book in $O(|V|)$ time.

- (1) Find an addition sequence G_1, G_2, \dots, G_m for G , with addition faces F_2, \dots, F_m .
- (2) Let $H=G_1$ be the initial superhamiltonian cycle.
- (3) For $k=2, 3, \dots, m$, execute steps 4.1 and 4.2.
 - (3.1) To match the context of the proof of Theorem 4.8, let $G^l=G_{k-1}$, let $H'=H$, let $G=G_k$ and let $F=F_k$.
 - (3.2) Construct H from H' as in the inductive step of the proof of Theorem 4.8.

Algorithm 4.3. Superhamiltonian Cycle of a Biconnected Trivalent Planar Graph

4.5. Oriented Face Traversal

Let G be a biconnected trivalent planar graph. Our task in this section is to show that an addition sequence for G can be constructed in linear time. We do this by constructing the faces F_m, F_{m-1}, \dots, F_2 corresponding to a subtraction sequence $G_m, G_{m-1}, \dots, G_2, G_1$ for G in linear time. Our strategy is to find a spanning tree in $G^I = G^D - \{U\}$, the *interior dual* of G , such that a particular traversal order of the tree yields the desired sequence F_m, F_{m-1}, \dots, F_2 . In particular, the first face in this traversal, F_m , must be a Class I face of G .

The key is the construction of the "proper" spanning tree for G^I . We observe that G^I is planar. Therefore, the planar embedding of G^I yields a circular (say counterclockwise) ordering of the edges of E^D incident to any $F \in V^D - \{U\}$. The generation of the spanning tree is similar to depth-first search except for two points. First, the counterclockwise ordering of the edges incident to F makes the choice of the order of edge traversal deterministic once the first face is chosen. Second, the tree is such that a post-order traversal yields a sequence of Class I faces (though they are Class I faces of successively smaller graphs). By Lemma 4.5, this second point is equivalent to the traversal yielding a sequence of faces that are not cutpoints of the untraversed subgraph of G^I .

Algorithm 4.4 generates the spanning tree T for G^I by *oriented face traversal*. It is important to recognize that F , F' and T are *variables* in this algorithm. In some general sense, Algorithm 4.3 traverses the faces in G^I from the outer face in. The traversal order is deterministic relative to the choice of K . When a face F first becomes "current" in step 4, exactly one of its incident edges e^D is marked "visited." The ordering of the remaining edges of G^I incident to F is consistently chosen in a counterclockwise direction from e^D . In step 5, the search for an incident edge that is not marked "visited" begins at e^D and proceeds in a counterclockwise direction. Step 5 makes the order of traversing edges incident to F deterministic (or oriented). A crucial difference between depth-first search and oriented face traversal occurs in step 8. Here the oriented face traversal fails to go "deeper" if an ancestor of the current face is encountered.

- (1) Let K be a boundary face of G .
- (2) If $G=K$, let $T=G^l$ and halt.
- (3) Let K' be the boundary face of G that is adjacent to K in a counterclockwise direction. Set $T = (\{K, K'\}, \{(K, K')\})$, and mark the edge (K, K') "visited."
- (4) Let $F=K'$. F is the "current" face in the traversal.
- (5) Let (F, F') be the first edge incident to F in a counterclockwise direction that is not marked "visited." If there is no such edge, and $F=K$, then halt. If there is no such edge, and $F \neq K$, then set F to its father, and repeat step 5.
- (6) Mark (F, F') "visited."
- (7) If F' is not in T , then add vertex F' and edge (F, F') to T , set F to F' , and go to step 5.
- (8) (F' is in T .) If F' is an ancestor of K in T , then set K to be its father, and go to step 5.
- (9) (F' is in T , and is not an ancestor of F .) Go to step 5 (F remains the same).

Algorithm 4.4. Oriented Face Traversal

Figure 4.15 illustrates an execution of Algorithm 4.4 on the graph of Figure 4.7. The resulting T is shown with solid lines. The interior faces are numbered F_1 through F_5 in the order in which they are encountered. Therefore, $K=F_1$. The interesting point is that, while F_3 is

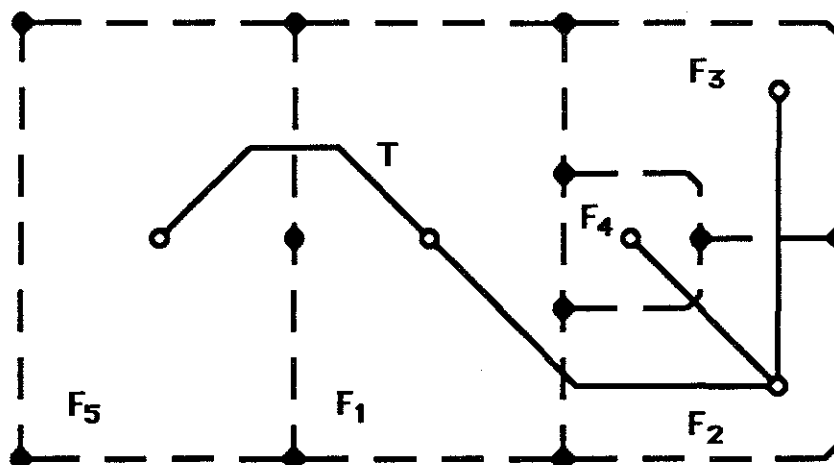


Figure 4.15. Oriented Face Traversal

adjacent to F_4 , the edge (F_3, F_4) is not added to T . This is because F_3 encounters its ancestor F_1 in step 8. F_4 is added to T when F_2 , the father of F_3 , is "current" for a second time.

We begin a sequence of lemmas that culminates in the statement that a post-order traversal of T yields a subtraction sequence for G .

Lemma 4.14. If T is a graph generated by Algorithm 4.4, then T is a tree.

Proof Step 3 initializes T to be a graph with two vertices and one edge. When step 7 adds an edge to T , it also adds a vertex to T . Hence the number of vertices in T is always one greater than the number of edges. Since T is clearly connected, T is a tree. \square

We prove that T is a spanning tree for G^I in Lemma 4.15. Before we can do that, we need two other lemmas. Call the first vertex F in T that encounters an ancestor in step 8 of Algorithm 4.4 the *leftmost* vertex of T . (If Algorithm 4.4 halts in step 2, call the sole vertex in T the leftmost.) Clearly, this F is a leaf of T . Furthermore, F is the first vertex encountered in a post-order traversal of the rooted, oriented tree T .

Lemma 4.15. If F is the leftmost vertex of the tree T generated by Algorithm 4.4, then F is a boundary face of G of Class I.

Proof Given the method of choosing the next edge to traverse in step 5, it is clear that Algorithm 4.4 stays in $BF(G)$ until an ancestor is encountered in step 8. Hence, F is a boundary face of G . Let F' be the ancestor of F encountered in step 8. Then there is a cycle in G^I consisting of the path in T from F to F' together with the edge (F, F') . Figure 4.16 illustrates the situation. The edges on the path (v_1, v_2, \dots, v_i) on the bounding cycle of F are in the interior of the cycle in G^I . Hence, (u_1, \dots, u_j) is the unique path in G_F . We conclude that F is a Class I face. \square

Lemma 4.16. Suppose G^I contains more than one vertex. Let T be the tree generated by Algorithm 4.4 when given G as input. Let F be the leftmost vertex of T , and let $G' = G - F$. Let T' be the tree generated by Algorithm 4.4 when given G' as input, assuming the same choice for K in step 1. Then $T' = T - \{F\}$.

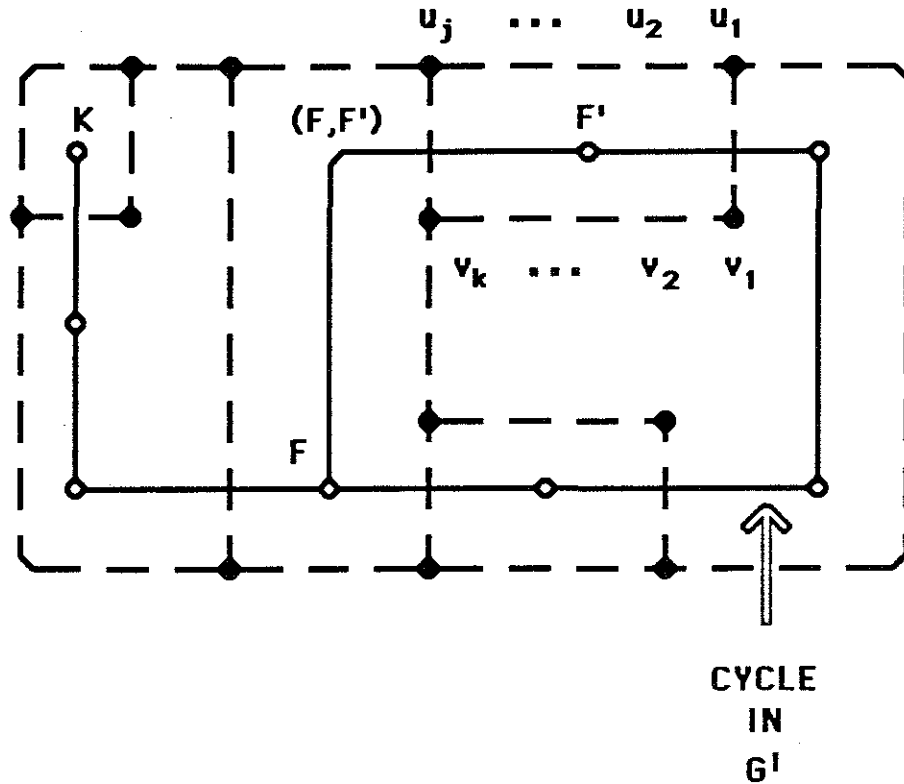


Figure 4.16. Proof of Lemma 4.15

Proof: Run Algorithm 4.4 on G until F is the current face. Since F is a leaf of T , the F' determined in step 5 is already in T , and the current face is changed to be the father of F in step 9. Since F is not the ancestor of any vertex of T , the remainder of the execution ignores any edge incident to F ; the presence of F has no further effect on T . Thus, removal of all parts of the execution of Algorithm 4.3 involving F yields the same result as the execution on G' . Therefore, $T \stackrel{!}{=} T - \{F\}$. \square

Lemma 4.17. T is a spanning tree of G' .

Proof: We must show that T spans G' . Let k be the number of interior faces of G . The proof is by induction on k . If G has one interior face, the result is clear.

For purposes of induction, assume that $k > 1$ and that the result is true for $k-1$ interior faces. Let F be the leftmost vertex of T . Let $G' = G - F$. Then G' has $k-1$ interior faces and the tree generated for G' is $T' = T - \{F\}$ (by Lemma 4.14). Hence, T' spans G' by inductive hypothesis. Therefore, T spans G . The lemma follows by induction. \square

We now have results indicating that oriented face traversal can be used to select a Class I face and that the deletion of that face leaves an oriented face traversal tree. It is important to note that Algorithm 4.4 executes in $O(|V|)$ time. A proof of this fact is similar to one for depth-first search. No edge of G^I is traversed more than twice (once in each direction). The only subtle point is the test for ancestor in step 8. It is essential that this test be done in constant time. Much as in depth-first search, this test can be accomplished by comparing the levels of faces F and F' in the tree T . Algorithm 4.4 can be augmented to store the level in T with each face in G^I and then can accomplish the test in constant time. Therefore, Algorithm 4.4 can be implemented in linear time.

The main result of this section is the construction of a subtraction sequence for G in linear time. Algorithm 4.5 accomplishes this. By Lemma 4.17, every interior face of G occurs in the sequence $F_m, F_{m-1}, \dots, F_2, F_1$. Define the sequence of graphs $G_m, G_{m-1}, \dots, G_2, G_1$ by $G_m = G$, $G_k = G_{k+1} - F_{k+1}$, $1 < k < m$ and $G_1 = G_F$. By induction on m and application of Lemmas 4.13 and 4.14, $G_m, G_{m-1}, \dots, G_2, G_1$ is a subtraction sequence for G . It is clear that Algorithm 4.5 can be implemented in linear time. Hence we have the following theorem.

- (1) Find an oriented face traversal tree T for G using Algorithm 4.3.
- (2) Traverse T in post-order and label the faces $F_m, F_{m-1}, \dots, F_2, F_1$ in the order visited.
- (3) F_m, F_{m-1}, \dots, F_2 defines the subtraction sequence for G .

Algorithm 4.5. Construction of a Subtraction Sequence

Theorem 4.18. Let G be a biconnected trivalent planar graph. Then a subtraction sequence for G can be constructed in linear time.

Corollary 4.19. Let G be a biconnected trivalent planar graph. Then a addition sequence for G can be constructed in linear time.

Proof: Take the addition sequence corresponding to the subtraction sequence produced by Algorithm 4.5. \square

This corollary completes the proof that Algorithm 4.1 executes in linear time.

4.6. Conclusions

In this chapter, we have shown that every trivalent planar graph is subhamiltonian, hence two-page embeddable. We have also given a linear time algorithm for constructing a two-page embedding of a trivalent planar graph. An obvious question is whether similar results are possible for higher valences. Define MV to be the largest integer such that all planar graphs of valence at most MV are subhamiltonian. The maximal planar graph that is not hamiltonian given in Capobianco and Molluzzo [CM] has valence eight. Hence seven is an upper bound for MV . Our result gives a lower bound of three for MV .

We point out some difficulties with extending our approach to show four to be a lower bound for MV . Most of our lemmas do not hold for quadrivalent planar graphs. In particular, the proof of Lemma 4.3 implies a special property of degree three vertices in a planar graph. If v is a degree three vertex, then the (at most three) faces adjacent to v are pairwise adjacent in the dual graph. The same cannot be said for a degree four vertex. This observation is the crux of the proof that BF is connected (Lemma 4.4). Thus it is likely that boundary faces would have to be redefined to include interior faces that share only isolated vertices with the exterior face. We are, therefore, not hopeful that our approach can be extended to valence four.

The upper bound of seven could be lowered by exhibiting a planar graph of valence at most seven that cannot be triangulated to obtain a hamiltonian graph. There is a great deal of freedom in the triangulation; the triangulation of each non-triangular face is independent of that of

all others. Therefore, a proof that all triangulations fail to produce a hamiltonian graph requires the consideration of many combinations.

We feel that oriented face traversal is of interest in its own right. It provides a traversal sequence for the dual graph such that, at each point of the traversal, the untraversed subgraph is connected. Our results on oriented face traversal extend to valences greater than three if a broader definition of boundary face is adopted. It would be interesting to find other applications for oriented face traversal.

CHAPTER 5

EMBEDDING OUTERPLANAR GRAPHS IN SMALL BOOKS

In this chapter, we consider tradeoffs between the pagenumber and pagewidth of book embeddings. The question of interest is: Given a graph G that admits a p -page embedding with pagewidth w , can G be embedded in a book of $p+c$ pages, where c is a small constant, with pagewidth significantly less than w ? Chung, Leighton and Rosenberg [CLR] present a sequence of outerplanar graphs $\{L_m\}$ for which the answer is in the affirmative. Each L_m has $2m$ vertices and requires pagewidth $\lceil m/2 \rceil$ in any one-page embedding but can be embedded in two pages with pagewidth 2. The main result of this chapter is an algorithm for embedding any d -valent n -vertex outerplanar graph in a two-page book with pagewidth $Cd \log n$, where $C = 8 / \left\lceil \log \frac{3}{2} \right\rceil$. This result is within a constant factor of optimal in pagewidth for the class of outerplanar graphs. Throughout this chapter, n denotes the number of vertices in the graph G .

5.1. Tradeoffs

We investigate the problem of tradeoffs between pagenumber and pagewidth in book embeddings. Motivation is best provided by an example from Chung, Leighton, Rosenberg [CLR]. The example is a sequence of outerplanar graphs $\{L_m\}$ for which any one-page embedding requires large pagewidth $\lceil m/2 \rceil$, but for which there exist two-page embeddings with pagewidth 2. The sequence consists of m -ladders (in [CLR], a m -ladder is called a *depth- m K_2 -cylinder*). The m -ladder L_m has vertex set

$$\{u_1, \dots, u_m\} \cup \{v_1, \dots, v_m\}$$

and edge set

$$\{(u_b, u_{k+1}) | 1 \leq k < m\} \cup \{(v_b, v_{k+1}) | 1 \leq k < m\} \cup \{(u_b, v_k) | 1 \leq k \leq m\}.$$

The first two components of the edge set constitute the two *sides* of the ladder while the last component constitutes its *rungs*. Figure 5.1 illustrates L_7 . The sides are solid and the rungs are dashed.

The m -ladder is clearly outerplanar and biconnected. By biconnectivity, L_m has a unique outerplanar embedding (Syslo [Sy]). Therefore, L_m has a unique one-page embedding up to reflection and circular permutation. Figure 5.2 illustrates a one-page embedding of L_7 of minimal pagewidth over all one-page embeddings. The rungs $\{(u_4, v_4), (u_5, v_5), (u_6, v_6), (u_7, v_7)\}$ nest over the interval (u_7, v_7) . Hence the pagewidth is ≥ 4 . A moment's reflection generalizes this observation: In any one-page embedding for L_m , at least $\lceil m/2 \rceil$ rungs nest over some interval; hence pagewidth is $\geq \lceil m/2 \rceil$.

Figure 5.3 illustrates a two-page embedding for L_7 that has pagewidth 2. The corresponding superhamiltonian cycle is illustrated in Figure 5.4. This superhamiltonian cycle is easily generalized, giving a two-page embedding of any L_m with pagewidth 2.

We now discuss tradeoffs in the general setting of an arbitrary graph G . Let P be the pagenumber of G . For each $p \geq P$, there exist one or more embeddings of G in a p -page book.

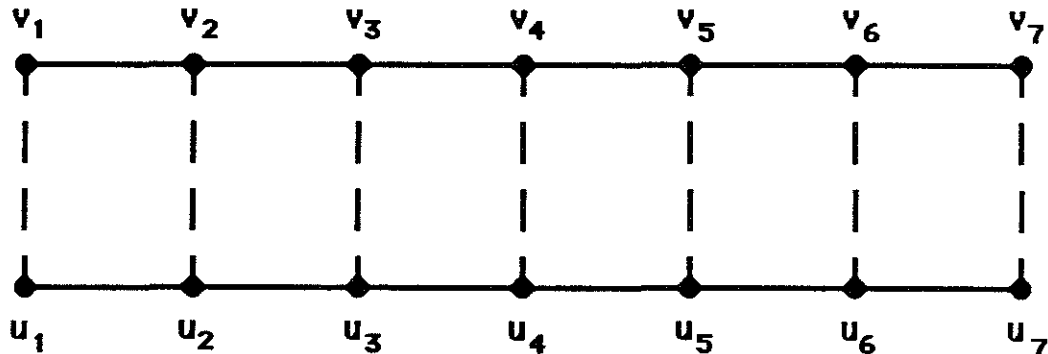


Figure 5.1. The 7-Ladder L_7

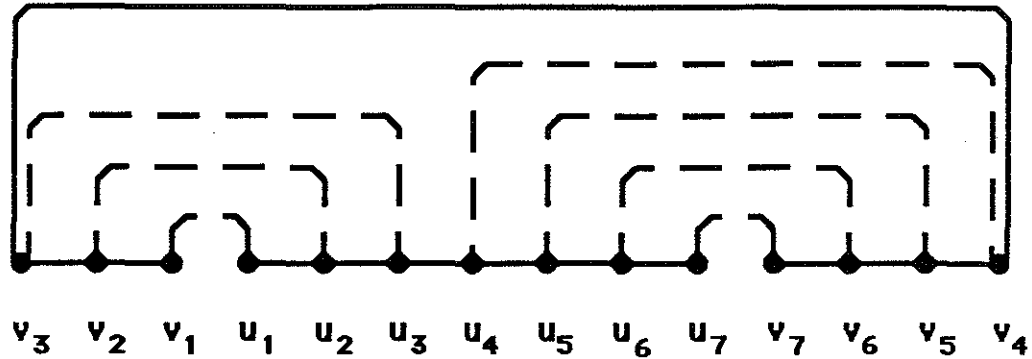


Figure 5.2. One-Page Embedding for L_7

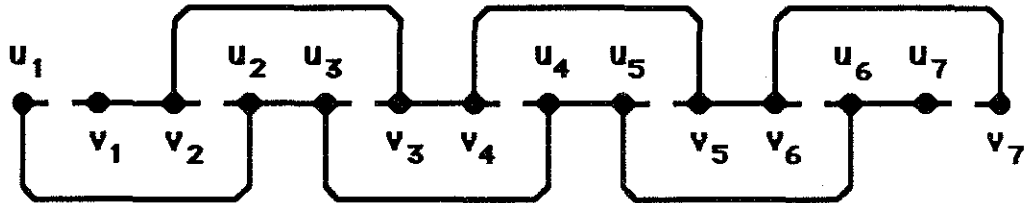


Figure 5.3. Two-Page Embedding for L_7

Among all those p -page embeddings of G , let w_p denote the pagewidth of one having minimum pagewidth. These pagewidths are non-increasing:

$$w_p \geq w_{p+1} \geq \dots \geq w_p, p \geq P.$$

In the extreme case that $p \geq |E|$, $w_p = 1$, as each edge may be assigned to a distinct page.

We are particularly interested in the product pw_p . We seek cases where pw_p is within a constant factor of the cutwidth of G . Note that pw_p is an upper bound on the cutwidth of the best p -page embedding of G . In the context of the DIOGENES approach discussed in the first chapter, pw_p is an upper bound on the height of a p -stack DIOGENES layout of G . Hence, we seek

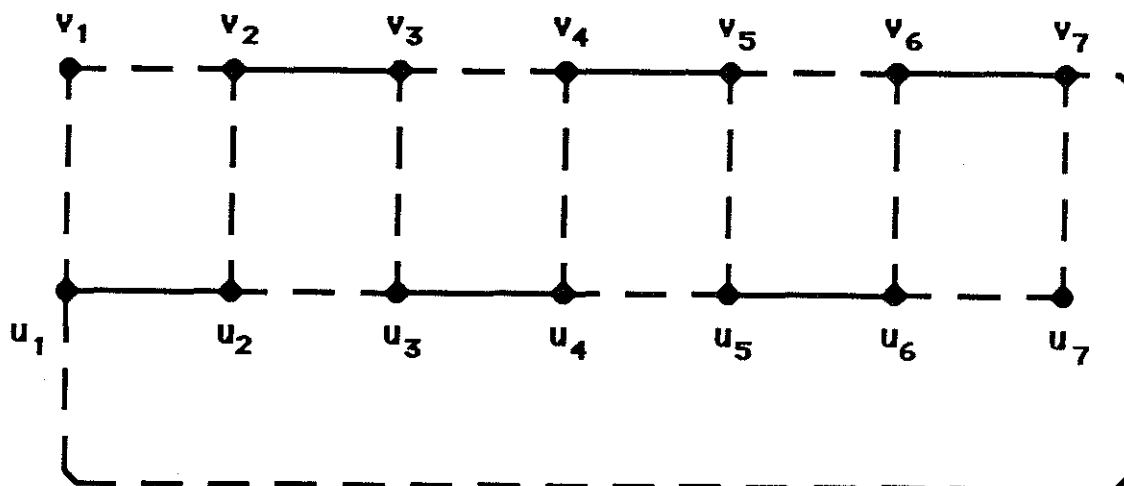


Figure 5.4. Superhamiltonian Cycle for L_7

DIOPHANTINE layouts of G that are within a constant factor of optimal in area over all linear layouts and within a small additive constant of optimal in stacknumber.

Our result is for the class of one-page (i.e., outerplanar) graphs. The m -ladder exhibits an extreme pagewidth tradeoff between one-page and two-page embeddings. For general outerplanar graphs, we do not expect such an extreme tradeoff. Since there exist outerplanar graphs that have one-page embeddings of minimal pagewidth, e.g. complete binary trees, the tradeoff in going from one page to two pages can be arbitrarily small, even zero.

An n -vertex complete $(d-1)$ -ary tree has cutwidth $\geq (d/2) \log n$ (Lengauer [Len]). (All logarithms are to the base 2.) Hence, any book embedding of a complete $(d-1)$ -ary tree in a constant number of pages requires pagewidth $\Omega(d \log n)$. In general, we cannot assume that outerplanar graphs have pagewidth $o(\log n)$.

5.2. Overview of the Algorithm

The tradeoff result we show is that any d -valent outerplanar graph G can be embedded in a two-page book with pagewidth $Cd \log n$, where $C = 8 / \left\lceil \log \frac{3}{2} \right\rceil$. From the observations in the preceding section regarding m -ladders and complete $(d-1)$ -ary trees, this result is optimal in pagewidth and within a constant factor of optimal in pagewidth for the class of d -valent outerplanar graphs. We prove our result via a recursive algorithm.

We aim for an algorithm that, when given an n -vertex d -valent outerplanar graph, returns a two-page embedding with pagewidth logarithmic in n . The input and output requirements of such an algorithm are a useful place to start.

The input to the algorithm is a d -valent outerplanar graph $G = (V, E)$. The manner of representing this input should witness the outerplanarity of G . Hence, a one-page embedding of G is the required form for the input. The linearization of V orders the vertices and provides names $1, 2, \dots, n$ for the vertices. The order of the vertices in a two-page embedding will *not* be the original order, but we shall continue to use the *names*. Since the algorithm is recursive, the same vertex will have different names at different levels of recursion. Figure 5.5 illustrates the form of the input when $G = L_7$.

The output of the algorithm is a two-page embedding of G with logarithmic pagewidth. To give a two-page embedding for G , it is sufficient to give a superhamiltonian cycle H in a supergraph G' of G (Proposition 2.5). $G' = (V, E)$ is actually a multigraph that contains all the edges of G plus possibly edges added to obtain H . $H \subseteq E$ is a set of n edges; since H is superhamiltonian, each of $1, \dots, n$ appears exactly twice among these edges. H represents $2n$ different book embeddings for G : there are n choices for the leftmost vertex, and there are two directions to the cycle. The algorithm fixes the desired book embedding by giving the pair of vertices (x, y) of the leftmost and rightmost vertices of the two-page embedding. We call x and y the *vertices of attachment* for G' . The output of the algorithm is then the ordered triple $(G', H, (x, y))$.

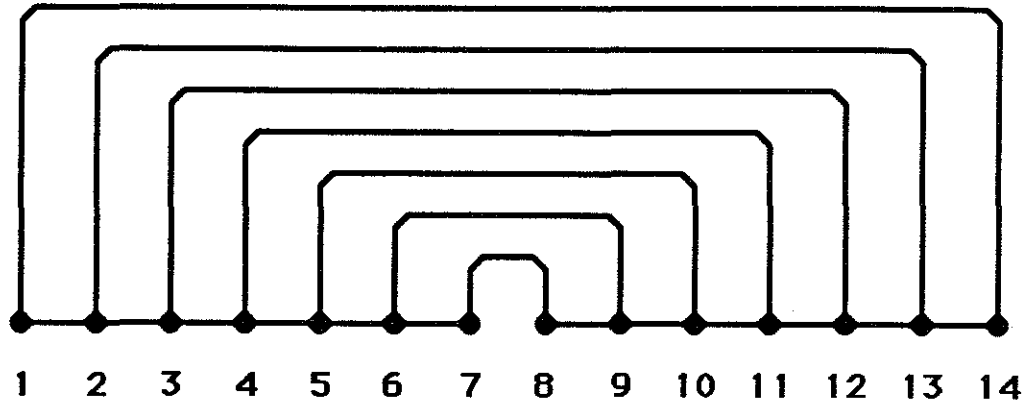


Figure 5.5. Input Representation for L_7

We imagine the one-page embedding of G as follows. The vertices are on a horizontal line in a plane, and the edges are arcs in the upper half-plane. In general, there are many sets of edges that can be added to G without destroying planarity. We restrict ourselves to two types of edges, *upper edges* and *lower edges*, depending on which half-plane the edges are embedded in. (Thus our restriction is that no edge uses both half-planes in its embedding.) The original edges of G are always upper edges. The algorithm may add an upper edge if it will not cross an existing upper edge. The algorithm may add a lower edge if it will not cross an existing lower edge. In particular, we may assume that the upper edge $(i, i+1), 1 \leq i < n$ is always present in G . If it is not already present, it can be added safely. Note that *all* added edges are removed at the end of the algorithm for purposes of determining pagewidth. Therefore, the added edge $(i, i+1)$ does not contribute to pagewidth (but an already present edge $(i, i+1)$ does).

The algorithm uses the divide-and-conquer paradigm. It determines subgraphs of G to work on separately before the results are joined together to obtain G' . Each subgraph is induced by a subinterval of $[1, n]$. We define a closed subinterval $[i, j]$ to be $\{i, i+1, \dots, j\}$. We define two types of half-closed, half-open subintervals: $[i, j)$ denotes $[i, j-1]$ and $(i, j]$ denotes $[i+1, j]$. For any subinterval α , $\text{size}\{\alpha\}$ denotes the number of vertices in the subinterval. Hence,

$\text{size}\{[i,j]\} = j - i + 1$. Define $G[i,j]$ to be the subgraph of G induced by the vertices in the interval $[i,j]$. If the algorithm is applied to $G[i,j]$ the result is $(G'[i,j], H(x,y))$, where (x,y) determines the first and last vertices of a two-page embedding of $G[i,j]$ with pagewidth logarithmic in $\text{size}\{[i,j]\}$.

The choice of subintervals depends on the structure of the one-page embedding of G . Define an *exposed vertex* w of G to be one for which G contains no (upper) edge (u,v) satisfying $u < w < v$. Thus an exposed vertex w is one that is "visible" from the infinite region of the upper half-plane. Each exposed vertex of G except 1 and n is a cutpoint of G whose removal separates G into left and right subgraphs.

An example will illustrate the divide-and-conquer paradigm. Figure 5.6 shows a sample G in a one-page embedding. The exposed vertices of G are 1, 3, 7 and 10. The algorithm recognizes that each of the edges $(1,3)$, $(3,7)$ and $(7,10)$ is "highest" in the sense that no other edge passes over it. These three edges determine three *nondisjoint* subintervals $[1,3]$, $[3,7]$ and $[7,10]$. In order to decompose the interval into *disjoint* subintervals, the algorithm chooses the largest, $[3,7]$, to remain intact, and removes one vertex from each of the other two subintervals. The resulting subintervals are $[1,2]$, $[3,7]$ and $[8,10]$. The algorithm recursively applies itself to each of the subintervals. The result to this point is shown in Figure 5.7. Each subproblem displays a superhamiltonian cycle of its subgraph and the first and last vertices of the corresponding two-

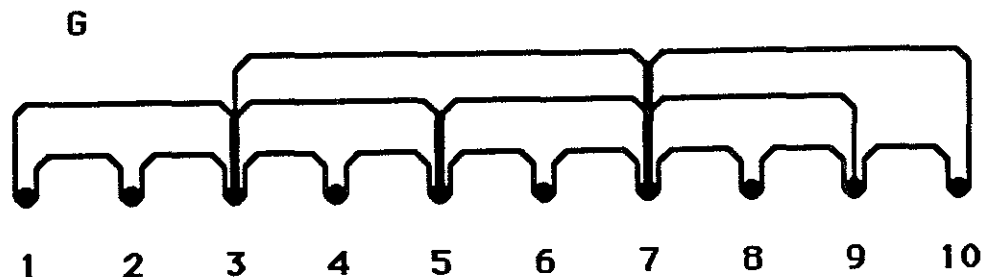


Figure 5.6. Sample G for Divide-And-Conquer

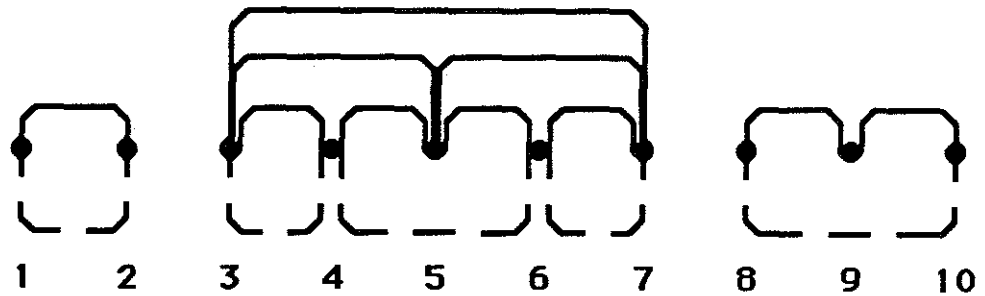


Figure 5.7. Results of Subproblems

page embedding. In Figure 5.8, these three superhamiltonian cycles are replaced by a superhamiltonian cycle for the entire graph. Lower edges $(1,2)$, $(4,6)$ and $(8,10)$ are deleted and lower edges $(2,4)$, $(6,8)$ and $(1,10)$ are added.

If two exposed vertices i and j are joined by an (upper) edge (i,j) , then there are no other exposed vertices in the interval $[i,j]$. In this case, we call $G[i,j]$ a *block*, denoted $B[i,j]$. When the

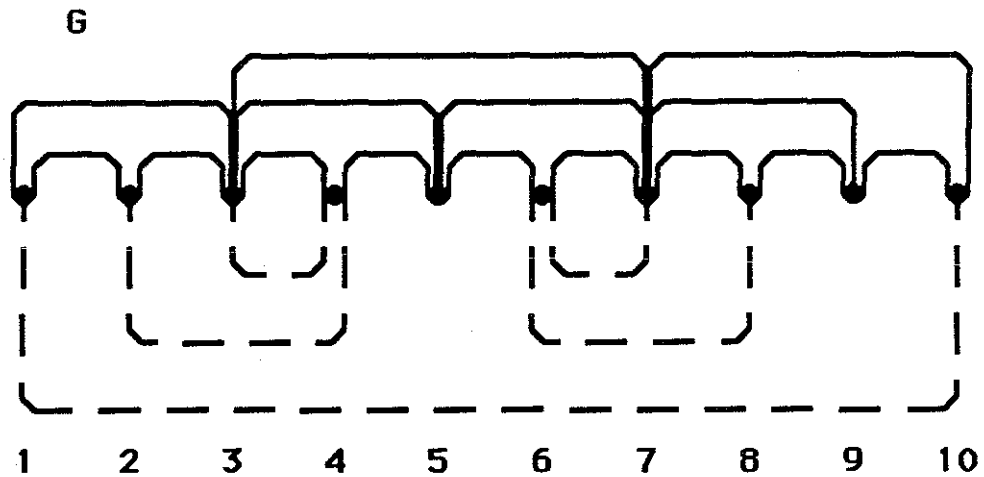


Figure 5.8. Superhamiltonian Cycle for G

interval $[1, n]$ is partitioned into subintervals, there will be edges with endpoints in different subintervals. Such *dangling* edges are exactly those edges of G not in any of the subgraphs generated by the subintervals. In the case of a block $B[i, j]$, these dangling edges can be incident to only i or j . The total number of such edges incident to i or j is called the *edge deficit* of $B[i, j]$, denoted $\text{def}\{[i, j]\}$. It is always true that

$$\text{def}\{[i, j]\} \leq 2(d-1).$$

In Figure 5.6, $B[1, 3]$, $B[3, 7]$ and $B[7, 10]$ are the blocks of G , and $\text{def}\{[3, 7]\} = 5$.

The example of Figures 5.6-5.8 illustrates the execution of the algorithm in the case that G has two or more blocks. There is another possible case: G has only one block. In that case, the divide-and-conquer construction is more complex. The two divide-and-conquer constructions corresponding to these two cases are developed in turn in the next two subsections.

5.2.1. String Construction

We now describe one of the two constructions used to obtain a superhamiltonian cycle for G from superhamiltonian cycles for the graphs induced by subintervals. It is called the *string construction*. (The name suggests that the superhamiltonian cycles for the subintervals are strung together sequentially to obtain a superhamiltonian cycle for the entire interval.) It is invoked when the number of exposed vertices is greater than two, so G is not one block. The partition into subintervals keys on the largest block, say $B[i, j]$. $B[i, j]$ is taken to be one of the subintervals. Note that not every block can be chosen in a partition into subintervals, since blocks share endpoints.

A precise description of the partition into subintervals requires more notation. Let m_1, m_2, \dots, m_q be the exposed vertices of G in ascending order. Suppose $B[m_b, m_{k+1}]$ is the largest block in G . Figure 5.9 illustrates the situation. The partition into $q-1$ subintervals is

$$\{[m_1, m_2], [m_2, m_3], \dots, [m_{k-1}, m_k], [m_b, m_{k+1}], [m_{k+1}, m_{k+2}], \dots, [m_{q-1}, m_q]\}.$$

Note that $B[m_b, m_{k+1}]$ is the only block of G in the partition. It is called the *key* block of the partition. The other subintervals in the partition are called *side* subintervals. Figure 5.10 illustrates the partition of G .

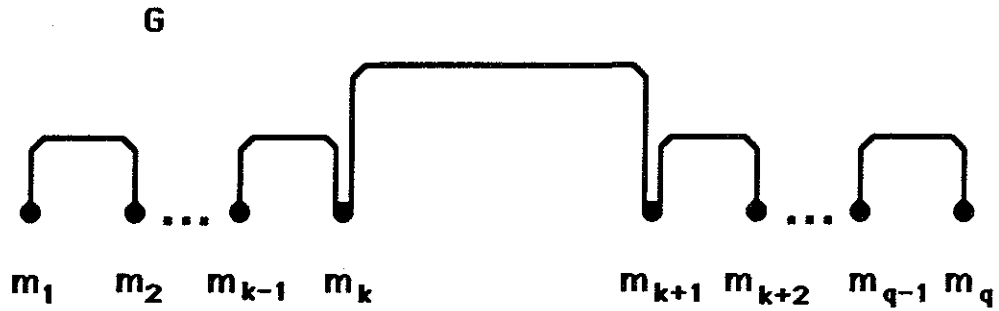


Figure 5.9. Exposed Vertices

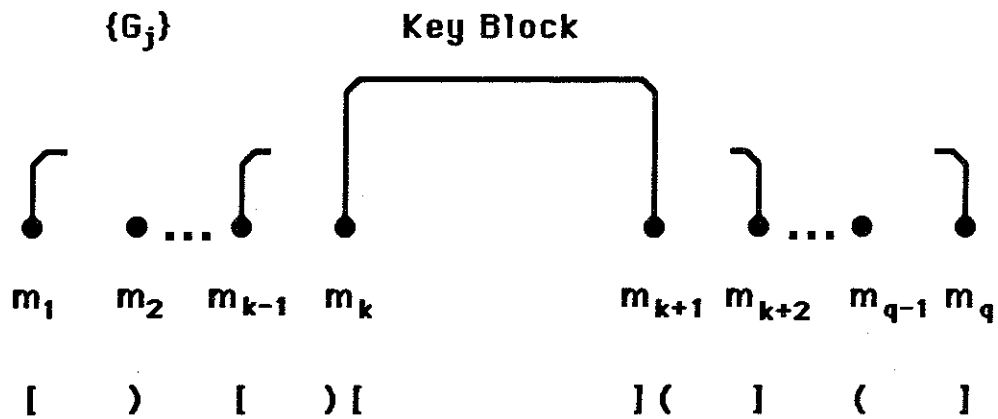


Figure 5.10. Partition Into Subintervals

The algorithm is recursively applied to the j th subinterval to obtain

$$(G'_j, H_j, (x_j, y_j)).$$

G' is obtained in two steps. First, all edges added to the $\{G_j\}$ are added to G (Figure 5.11).

Second, the lower edges $\{(x_j, y_j)\}$ are deleted and the lower edges

$$\{(y_j, x_{j+1}) | 1 \leq j \leq q-2\} \cup \{(x_1, y_{q-1})\}$$

are added. H is obtained from $\bigcup H_j$ by deleting and adding the same edges (Figure 5.12). Assign-

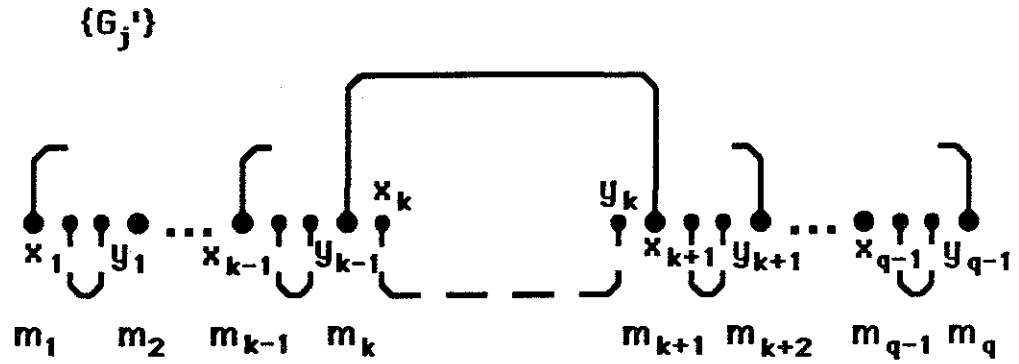


Figure 5.11. Results for Subintervals

ing $(z, y) = (x_1, y_{q-1})$ completes the string construction. The correctness of the construction is proved in Lemma 5.3.

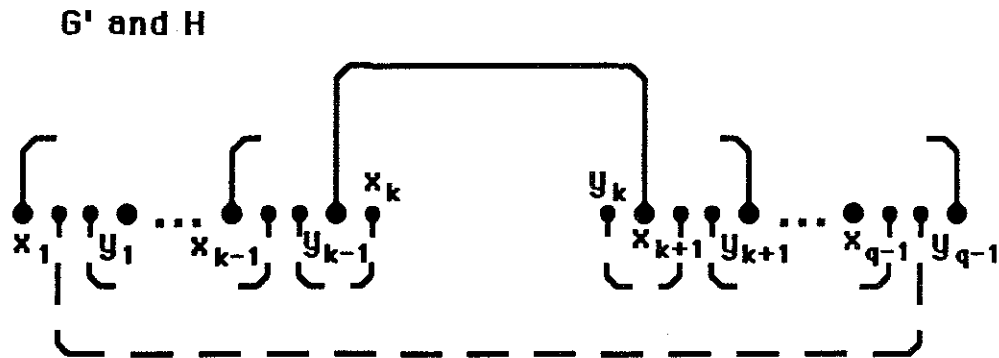


Figure 5.12. Subintervals Strung Together

5.2.2. Ladder Construction

In this subsection, we consider the case when G has only one block. We are unable to divide G into subintervals based on blocks. To reach a solution, we first focus on the problem of obtaining logarithmic pagewidth. To obtain logarithmic pagewidth, it is clearly sufficient that the linear layout corresponding to the two-page embedding have logarithmic cutwidth. An approach to small cutwidth is the recursive application of a *separator theorem* (see Lipton and Tarjan [LT]). A separator theorem states that the removal of some number of vertices from a graph will partition the remainder of the graph into two subgraphs of approximately equal size. For outerplanar graphs, a two-vertex separator always exists.

Lemma 5.1. Let G be an outerplanar graph containing at least 3 vertices. There exist vertices x and y whose removal separates G into subgraphs G_1 and G_2 such that $\frac{1}{3}n < |G_k| < \frac{2}{3}n, k=1,2$. If (x,y) is not an upper edge of G , then it can be added to G as an upper edge without inducing a crossing.

Proof: Since G is outerplanar, we can use the circular formulation of book embedding to embed it in a circle. The vertices of G are placed equidistantly on the circle. The edges of G are chords of the circle with no two chords intersecting. If the center of the circle lies on an edge, let x and y be the endpoints of the edge; the result follows. Otherwise, let F be the face the center is in. If two vertices on F are on a diameter, let them be x and y , and the result follows. Otherwise, triangulate F within the circle. The center of the circle lies within some resulting triangle (u,v,w) . We may assume that the angle $\angle uvw$ is the largest of the triangle. This angle is between 60° and 90° . Let $x=u$ and $y=w$. Let G_1 be the graph induced by the vertices within the angle $\angle uvw$, and let G_2 be the graph induced by the vertices outside the angle $\angle uvw$. Then the removal of x,y separates G into G_1 and G_2 where

$$\frac{1}{3}n < |G_1| < \frac{1}{2}n.$$

The lemma follows. \square

If (x, y) is not already an edge of G , it can be added without destroying outerplanarity. An edge (x, y) that satisfies Lemma 5.1 is called a *separating edge*. An algorithm to obtain logarithmic cutwidth for a d -valent outerplanar graph G can select a separating edge (x, y) and apply itself recursively to the resulting G_1 and G_2 . However, it is unclear how to obtain a superhamiltonian cycle for G from superhamiltonian cycles for G_1 and G_2 .

Our algorithm uses separating edges in another way so as to make it possible to derive a superhamiltonian cycle from superhamiltonian cycles for the pieces. The key is the following definition. Let G be an outerplanar graph, and let (x, y) be a separating edge for G . A set $P \subseteq E$ is *parallel* to (x, y) if

- (1) $(x, y) \in P$;
- (2) if $(u, v), (w, z) \in P$, then $\{u, v\} \cap \{w, z\} = \emptyset$ (there are no shared endpoints);
- (3) P can be ordered as $\{(u_1, v_1), (u_2, v_2), \dots, (u_t, v_t)\}$ in such a way that

$$u_1 < u_2 < \dots < u_t < v_t < \dots < v_2 < v_1$$

(the edges of P nest).

A sample set of parallel edges for a graph G is shown in Figure 5.13 by dashed lines. A set P of parallel edges is *maximal* if no edge of G can be added to P to obtain a larger set of parallel edges.

Suppose P is a maximal set of parallel edges for G . Let V_P be the set of endpoints of edges in P . The removal of the vertices V_P from G separates the interval $[1, n]$ into some number of subintervals. Let G_P be the subgraph of G resulting from the removal of V_P and all incident edges. Let $[i_1, j_1], \dots, [i_s, j_s]$ be these subintervals in left-to-right order. The planarity of G and the maximality of P guarantees that there is no edge of G between two vertices in different subintervals. This in turn guarantees that G_P can be obtained an alternate way: G_P is the (disjoint) union of the induced subgraphs $G[i_k, j_k], 1 \leq k \leq s$. By Lemma 5.1, the presence of a separating edge in P guarantees that

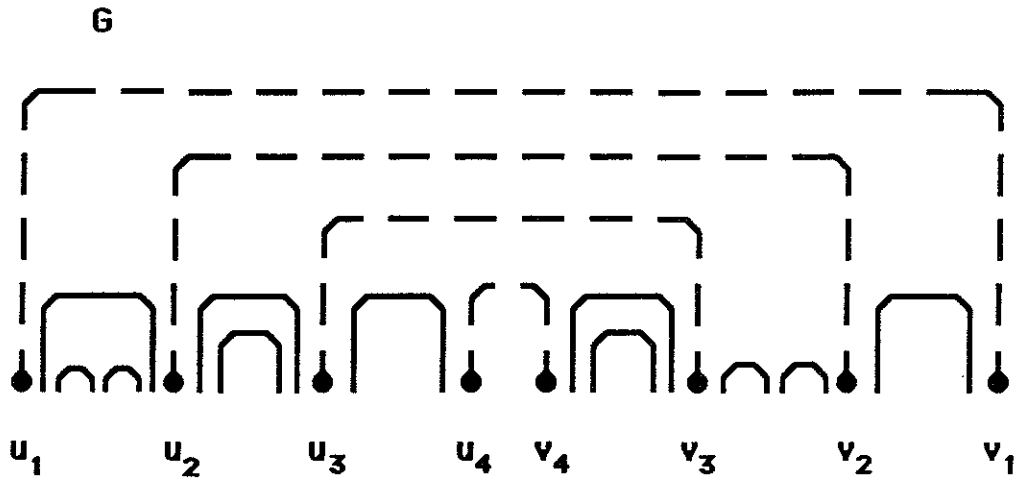


Figure 5.13. Parallel Edges in G

$$\text{size}\{[i_k, j_k]\} < \frac{2}{3}n, 1 \leq k \leq s.$$

Figure 5.14 shows the graph of Figure 5.13 after the removal of V_P .

The algorithm is applied recursively to each $G[i_k, j_k]$ to obtain a superhamiltonian cycle for each. To obtain a superhamiltonian cycle for G , one need only reintroduce the endpoints of the parallel edges V_P . A second look at Figure 5.13 provides inspiration. If each subinterval $[i_s, j_s]$ were replaced by an edge (i_{s-1}, j_s+1) between two vertices in V_P , the result is the one-page embedding of a ladder where *all* the rungs nest. The construction of a superhamiltonian cycle H for G is

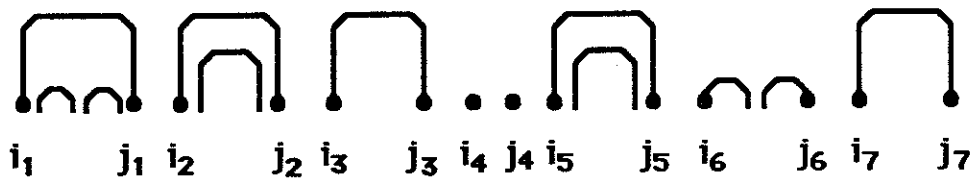


Figure 5.14. Removal of V_P

patterned after the superhamiltonian cycle for a ladder, as illustrated in Figure 5.4.

Appropriately, we name the construction of H the *ladder construction*. There are two cases to consider, depending on whether or not the edge $(1,n)$ is in P . The case $(1,n) \in P$ illustrates all the ideas and is simply modified to cover the case $(1,n) \notin P$.

Start with the picture of the parallel edges alone in Figure 5.15. Some lower edges are added to obtain a supercycle containing exactly the vertices in V_P . This supercycle is indicated in Figure 5.16 by arrows. It remains to place all the subintervals within this supercycle. To accomplish this, each lower edge is replaced by new lower edges that connect two subintervals into its place in the supercycle. For a right arrow (u_b, u_{k+1}) , the result is as in Figure 5.17. For a left arrow (v_{k-1}, v_t) , the result is as in Figure 5.18; t is chosen so that $[i_t, j_t]$ is the subinterval between v_{k+1} and v_t .

For the case $(1,n) \notin P$, $[i_1, j_1]$ is to the left of the ladder and $[i_s, j_s]$ is to the right of the ladder. The connection of $[i_1, j_1]$ into H is shown in Figure 5.19. The connection of $[i_s, j_s]$ into H is shown in Figure 5.20.

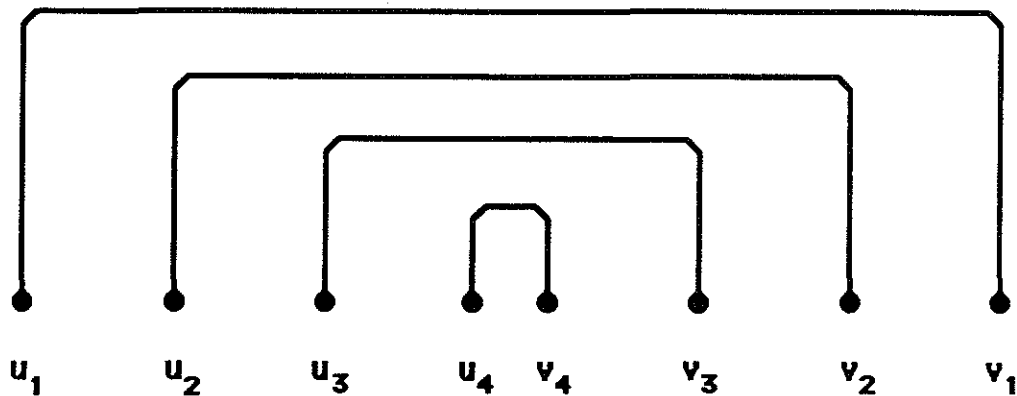


Figure 5.15. Parallel Edges

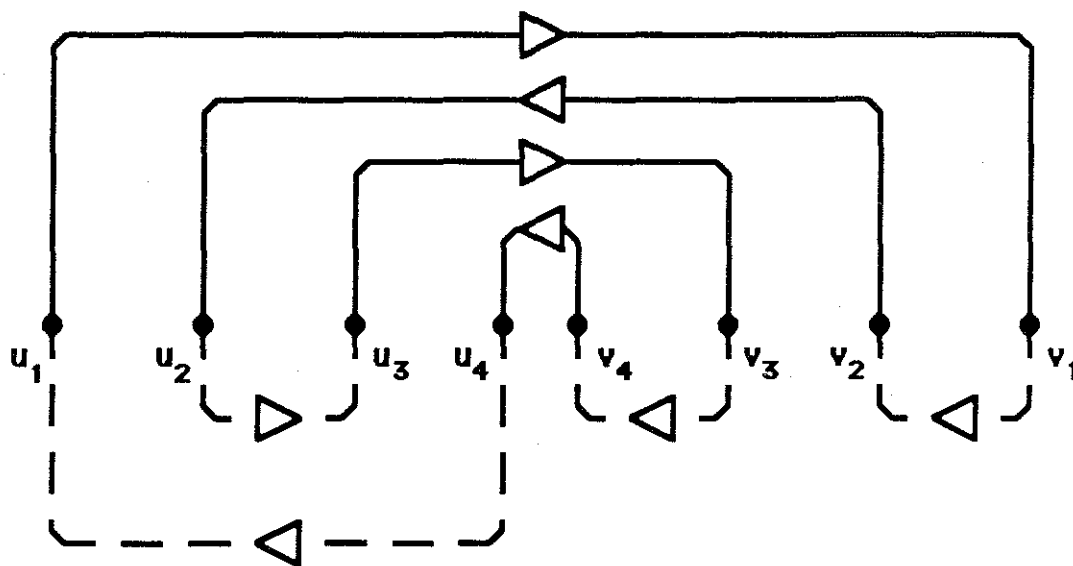


Figure 5.16. Supercycle for Parallel Edges

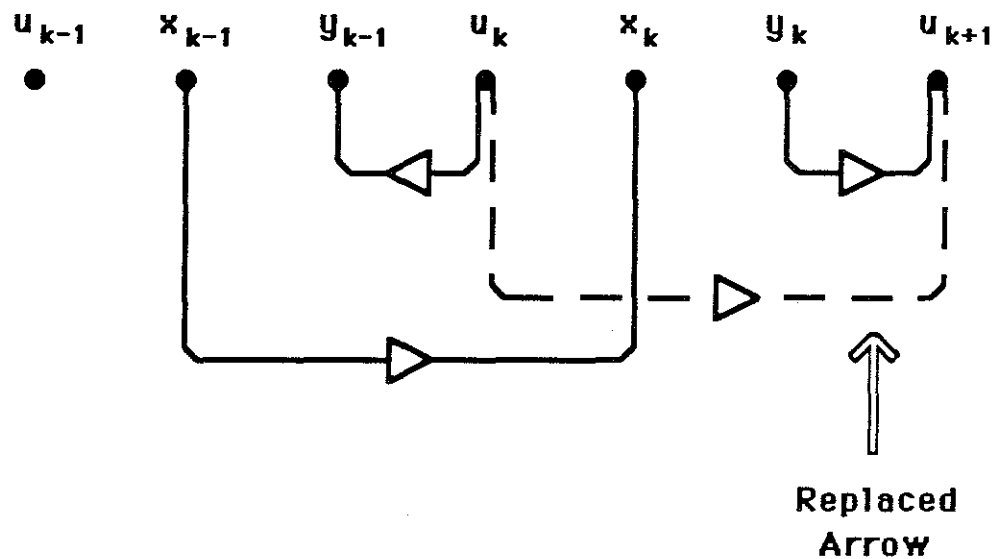


Figure 5.17. Replacing a Right Lower Edge

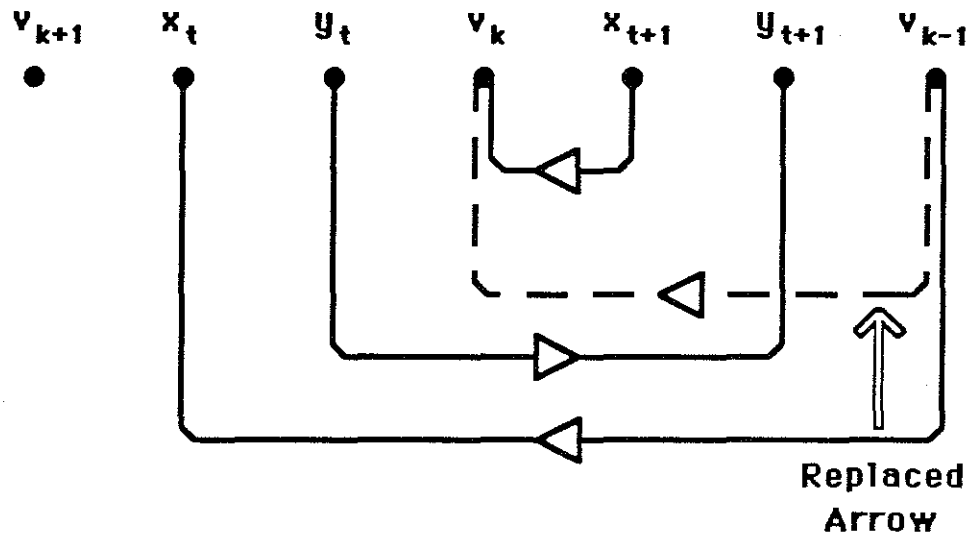


Figure 5.18. Replacing a Left Lower Edge

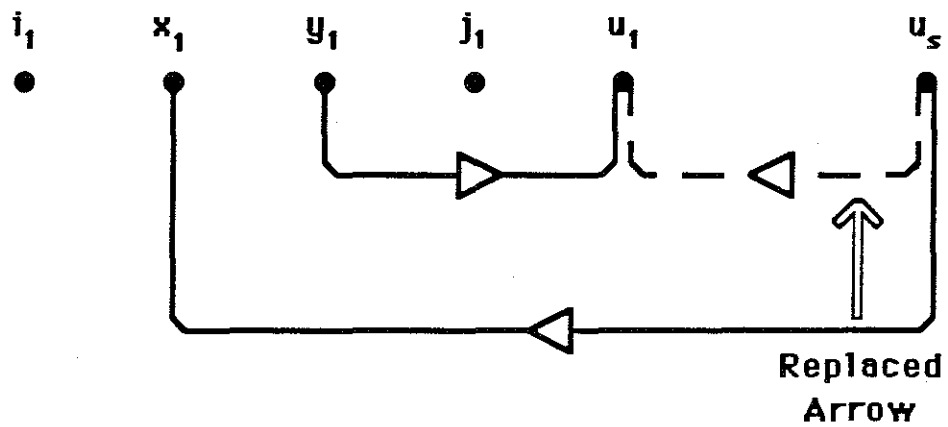


Figure 5.19. Adding a Subinterval on the Left

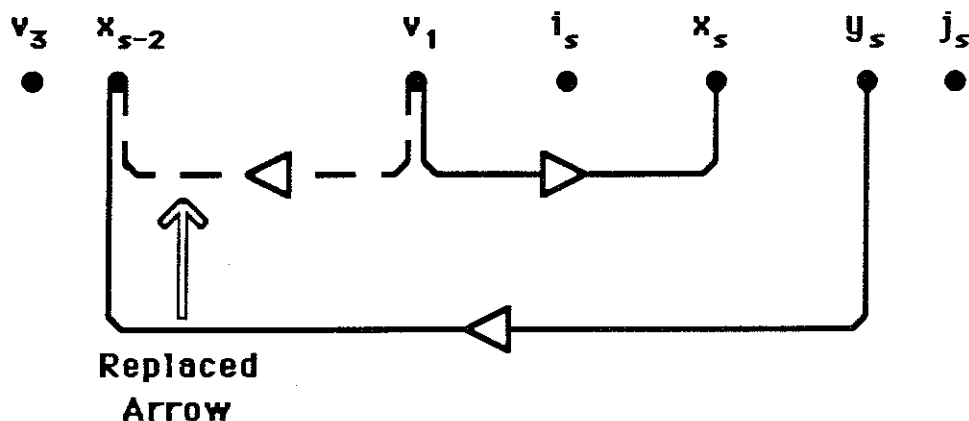


Figure 5.20. Adding a Subinterval on the Right

5.3. The Algorithm

This section describes our algorithm for embedding a d -valent outerplanar graph in a two-page book with logarithmic pagewidth. The correctness of the algorithm, embodied in Theorem 5.2, is given in the next section. Section 5.5 analyzes the performance of the algorithm.

The statement is Algorithm 5.1. As the algorithm is recursive, it is useful to give it a name. The name is TRADEOFF. TRADEOFF is a recursive function which has as input the d -valent outerplanar graph G and as output the planar supergraph G' having hamiltonian cycle H and vertices of attachment x and y .

For simplicity, it is to be noted that certain trivial cases are not included in the statement of TRADEOFF. These cases occur when a recursive invocation of TRADEOFF returns an empty G' . This cannot occur in step 5, as each subinterval contains at least one vertex. However, it can occur in step 9 when some G_k is empty. In that case, the ladder construction merely skips the empty interval $[i_k, j_k]$ (which is caused by two adjacent elements of V_p).

We now describe TRADEOFF step by step.

(1) These are the trivial cases when $n \leq 2$. If G is empty, return $G' = \emptyset$. If G is a single vertex, return G' having a single loop. If G has two vertices, then it has one edge $(1,2)$. Return G'

Function TRADEOFF(G), returns $(G', H, (x, y))$.

- (1) (Trivial cases)
 If $G = \emptyset$, then assign $G' = \emptyset$, $H = \emptyset$ and $(x, y) = \text{undefined}$.
 If $V = \{1\}$, assign $G' = (\{1\}, \{(1,1)\})$, $H = \{(1,1)\}$ and $(x, y) = (1,1)$.
 If $V = \{1,2\}$, assign $G' = (\{1,2\}, \{(1,2), (1,2)\})$, $H = \{(1,2), (1,2)\}$ and $(x, y) = (1,2)$.
 Return $(G', H, (x, y))$.
- (2) Let $S = \{m_j \mid 1 \leq j \leq q\}$ be the set of exposed vertices of G in increasing order.
- (3) Choose $k, 1 \leq k \leq q-1$ such that $B[m_k, m_{k+1}]$ is the key block of G .
- (4) If $B[m_k, m_{k+1}] = G$, then go to step 7.

String Construction

- (5) (G has more than one block.) For $1 \leq j < k$, assign

$$(G'_j, H_j, (x_j, y_j)) = \text{TRADEOFF}(G[m_j, m_{j+1}]).$$
 For $j = k$, assign

$$(G'_k, H_k, (x_k, y_k)) = \text{TRADEOFF}(G[m_k, m_{k+1}]).$$
 For $k < j < q$, assign

$$(G'_j, H_j, (x_j, y_j)) = \text{TRADEOFF}(G[m_j, m_{j+1}]).$$
- (6) Use the string construction to obtain G' , H and (x, y) . Return $(G', H, (x, y))$.

Ladder Construction

- (7) ($B[m_k, m_{k+1}] = G$.) Choose a separating edge (u, v) for G . If (u, v) is not already an edge of G , then add it as an upper edge.
- (8) Choose P a maximal set of edges parallel to (u, v) . Let V_P be the set of endpoints of edges in P .
- (9) $V - V_P$ determines a sequence of disjoint subintervals $[i_1, j_1], [i_2, j_2], \dots, [i_s, j_s]$. For $1 \leq k \leq s$, make the assignment:

$$(G'_k, H_k, (x_k, y_k)) = \text{TRADEOFF}(G[i_k, j_k]).$$
 Construct G' from G and $\{G'_k\}$ using the ladder construction. Return $(G', H, (x, y))$.

Algorithm 5.1. The Tradeoff Algorithm

having the added lower edge $(1,2)$ which is parallel to the upper edge $(1,2)$.

(2) From the one-page embedding of G , determine the exposed vertices of G . It is straightforward to accomplish this step in linear time with Algorithm 5.2. Algorithm 5.2 requires time $O(dn)$ and generates the elements of S in increasing order.

(3) Choose the key block of G , $B[m_k, m_{k+1}]$. Clearly, this can be accomplished in time linear in $|S|$.

(4) This step determines which of two cases is current. If G is a single block, then the ladder construction is applied (steps 7 through 9). If G has more than one block, then the string construction is applied (step 5 and 6).

(5) Decompose the interval $[1,n]$ into subintervals so that the key block $B[m_k, m_{k+1}]$ is one of the subintervals. Note that each side subinterval contains fewer than $\frac{1}{2}n$ vertices. Apply TRADEOFF to the graphs induced by each subinterval to obtain supergraphs $G_j, 1 \leq j \leq q-1$.

(6) Apply the string construction to obtain the planar hamiltonian supergraph G' and the hamiltonian cycle H for G' . Assign $(x,y) = (x_i, y_i)$. Return $(G', H, (x,y))$.

(7) We know that G is entirely covered by the edge $(1,n)$. We show that it is then safe to add a separating edge to G . If this is the initial call to TRADEOFF, we can always add a separating edge. If this is a deeper recursive call to TRADEOFF, we imagine that there are intervals to the left and right of $[1,n]$ with dangling edges incident to vertices in $[1,n]$. Since these

- (1) Assign $S = \{1\}$ and $i = 1$.
- (2) If $i > n$, then halt.
- (3) Assign $i = \max\{i+1, \max_{(i,k) \in E} k\}$.
- (4) Assign $S = S \cup \{i\}$. Go to step 2.

Algorithm 5.2. Determining Exposed Vertices in Linear Time

- (1) Triangulate the interior faces of G .
- (2) Examine each edge (u,v) of the triangulated G to find one such that $\frac{1}{3}n \leq (v-u) \leq \frac{2}{3}n$.

Algorithm 5.3. Finding a Separating Edge

dangling edges can only be incident to exposed vertices (in this case, 1 and n), any upper edge added to G at this recursive level cannot cross an edge at a higher recursive level. The determination of a suitable separating edge is accomplished in linear time by Algorithm 5.3. (Note that the triangulated G has a linear number of edges.)

(8) Select a maximal set of parallel edges. The construction of P is accomplished in linear time by Algorithm 5.4.

(9) This step completes the ladder construction. TRADEOFF is invoked recursively for each subinterval disjoint from V_P . G' and H are obtained by the ladder construction described in the previous section.

- (1) Assign $P = \{(u,v)\}$, $s = u-1$ and $t = v$.
- (2) If $s < 1$, then go to step 4.
- (3) Assign $r = \max\{1, \max_{(s,t) \in E} k\}$. If $r \leq t$, then assign $s = s-1$ and go to step 2. Else assign $P = P \cup \{(s,r)\}$, $s = s-1$ and $t = r$ and go to step 2.
- (4) Assign $s = u+1$, and $t = v$.
- (5) If $s \geq t$, then halt.
- (6) Assign $r = \max\{1, \max_{(s,t) \in E} k\}$. If $r > s$, then assign $P = P \cup \{(s,r)\}$, $s = s+1$ and $t = r$ and go to step 5. Else, assign $s = s+1$ and go to step 5.

Algorithm 5.4. Generating a Maximal Set of Parallel Edges

5.4. Correctness

In this section, we demonstrate the correctness of algorithm TRADEOFF. Correctness is embodied in the following theorem.

Theorem 5.2. Let G be a d -valent outerplanar graph. Let $(G', H, (x, y))$ result from applying TRADEOFF to G . Then H is a superhamiltonian cycle for G with the following property: following H from x to y yields a two-page embedding of G with pagewidth $\leq Cd \log n$, where C is a constant. C can be chosen to have any value $\geq 8 / \left\lceil \log \frac{3}{2} \right\rceil$.

Proof: The proof naturally decomposes into the proof of pagenumber (Lemma 5.3) and the proof of pagewidth (Lemma 5.4). \square

Lemma 5.3. Given the assumptions of Theorem 5.2, following H from x to y yields a two-page embedding of G .

Proof: The proof is by induction on n . The inductive hypothesis is:

(H.1) $G \subset G'$;

(H.2) G' is planar;

(H.3) H is a hamiltonian cycle of G' ;

(H.4) $(x, y) \in H$ is a lower edge of G' such that there is no lower edge (u, v) of G' with $u < x \leq y < v$ (i.e., x and y are on the unbounded region of the lower half-plane).

Step 1 of Algorithm 5.1 guarantees that the inductive hypothesis is satisfied when $n \leq 2$.

For purposes of induction, assume that the inductive hypothesis is true for graphs of size less than n and that $n > 2$. There are two cases determined by the cardinality of the set S of exposed vertices of G : (1) $|S| > 2$ and (2) $|S| = 2$.

(1) $|S| > 2$. TRADEOFF applies the string construction in steps 5 and 6. The inductive hypothesis guarantees that after the applications of TRADEOFF to all the subintervals, each x_j and each y_j is on the unbounded region of the lower half-plane. Therefore, the lower edges $(y_j, x_{j+1}), 1 \leq j \leq q-1$ and (x_1, y_q) can be added while maintaining planarity (H.2). Clearly, $G \subset G'$

(H.1), and H is a hamiltonian cycle of G' (H.3). Finally $x=x_1$ and $y=y_1$ satisfy H.4.

(2) $|S|=2$. The edge $(1,n)$ is in G and covers all other upper edges. TRADEOFF applies the ladder construction to G in steps 7 through 9. In section 5.2.2, the addition of the separating upper edge (u,v) was shown to maintain planarity. In step 9, the application of TRADEOFF to each $G[i_k, j_k]$ yields $(G'_k, H_k(x_k, y_k))$ that satisfies the inductive hypothesis. In particular, H.4 applies to each (x_k, y_k) . Since each x_k and y_k is on the unbounded region of the lower half-plane, the ladder construction yields a planar result (H.2). The ladder construction also makes $G \subset G'$ (H.1) and H a hamiltonian cycle of G' (H.3). Finally, (z,y) is explicitly chosen to satisfy H.4.

This extends the induction for arbitrary G . Since H is a hamiltonian cycle of a planar supergraph of G , it yields a two-page embedding of G [BK]. \square

To complete the proof of Theorem 5.2, we must bound the pagewidth of the two-page embedding. It is sufficient to bound the cutwidth of the underlying linear embedding. We use the notation $\text{cw}(H)$ to mean the cutwidth of the linear embedding obtained by following H from x through y . (G , x and y will be clear from context.) If i and j are vertices in H such that i comes before j in the linear embedding, define $\text{cw}([i,j])$ to be the cutwidth of the linear subembedding from i to j .

Lemma 5.4. Given the assumptions of Theorem 5.2, $\text{cw}(H) \leq Cd \log n$, where $C = 8 \left\lceil \log \frac{3}{2} \right\rceil$.

Proof: The proof is by induction on n . The statement of the inductive hypothesis mirrors the two cases of the algorithm. The inductive hypothesis is:

(I.1) If G has more than one block, then $\text{cw}(H) < Cd \log n$;

(I.2) If G is a single block, then $\text{cw}(H) \leq \max(1, Cd \log n) - \text{def}\{[1,n]\}$. Some explanation of the presence of the edge deficit in I.2 is in order. In the string construction, a large key block $[m_b, m_{b+1}]$ must be able to absorb $\text{def}\{[m_b, m_{b+1}]\}$ additional cutwidth, as its cutwidth will dominate the cutwidth of the entire string construction. The precise meaning of this statement will be clear from the proof. The $\max(1, Cd \log n)$ takes care of the case $n=1$. Note that a G with a single vertex can never be the key block in a string construction.

For the basis of the induction, it is easy to check the inductive hypothesis for $n=1$ and $n=2$.

For purposes of induction, assume that the inductive hypothesis is true for graphs of size less than n and that $n > 2$. There are two cases: (1) G has more than one block and (2) G is a single block.

(1) G has more than one block. In this case, the string construction is applied (steps 5 and 6). Let us examine the linear order induced by H and (x, y) on V . H_1 is a superhamiltonian cycle for $G([m_1, m_2])$ that begins at $x=x_1$ and ends at y_1 . As such, H_1 can be viewed as a permutation on $[m_1, m_2]$. The string construction places the vertices of $[m_1, m_2]$ first in H , in this permuted order. Similarly, the vertices of $[m_2, m_3]$ come next in H , in the permuted order given by H_2 . In general, the $q-1$ subintervals appear in the same order in H as they do in the partition, though H permutes the vertices within each subinterval. The permutation of the j th subinterval is always that of H_j .

It is now possible to bound $\text{cw}(H)$ based on $\{\text{cw}(H_j)\}$. First, consider the cutwidth of H between two subintervals, that is, $\text{cw}([y, x_{j+1}]), 1 \leq j \leq q-2$. Suppose $j < k$. Then the only edges that pass over the interval $[y, x_{j+1}]$ are dangling edges from m_{j+1} to $[m_j, m_{j+1})$. Hence,

$$\text{cw}([y, x_{j+1}]) \leq d-1 < Cd \log n.$$

If $j \geq k$, by a similar argument, we have

$$\text{cw}([y, x_{j+1}]) \leq d-1 < Cd \log n.$$

Second, consider the cutwidth over a side subinterval. Consider the j th subinterval in H , $[x, y_j]$. If $j < k$, then there are at most $(d-1)$ dangling edges from m_{j+1} to $[m_j, m_{j+1})$ that can contribute to $\text{cw}([x, y_j])$ and at most d dangling edges from m_j to $[m_{j-1}, m_j)$ that can contribute to $\text{cw}([x, y_j])$. Hence, by I.1,

$$\begin{aligned} \text{cw}([x, y_j]) &< 2d + Cd \log(\text{size}\{[m_j, m_{j+1}]\}) \\ &< Cd \log n \end{aligned}$$

since $\text{size}\{[m_j, m_{j+1}]\} < \frac{1}{2}n$. If $j > k$, we have similarly

$$\begin{aligned} \text{cw}([x, y]) &< 2d + Cd \log(\text{size}\{m, m_{k+1}\}) \\ &< Cd \log n. \end{aligned}$$

Third and finally, consider the cutwidth over the key block, $B[m_b, m_{k+1}]$. By 1.2,

$$\text{cw}([x_k, y_k]) \leq (Cd \log(\text{size}\{m_b, m_{k+1}\})) - \text{def}\{m_b, m_{k+1}\}.$$

The only dangling edges that can contribute to the cutwidth over $[x_k, y_k]$ are those incident to m_k and m_{k+1} . There are $\text{def}\{m_b, m_{k+1}\}$ of these. Hence

$$\begin{aligned} \text{cw}([x_k, y_k]) &\leq Cd \log(\text{size}\{m_k, m_{k+1}\}) \\ &< Cd \log n. \end{aligned}$$

Putting these three results together yields $\text{cw}(H) < Cd \log n$. Thus G satisfies 1.1.

(2) G is a single block. In this case, the ladder construction is applied (steps 7 through 9).

The subintervals are $[i_1, j_1], \dots, [i_s, j_s]$. Let $P = \{(u_1, v_1), (u_2, v_2), \dots, (u_t, v_t)\}$ where

$$u_1 < u_2 < \dots < u_t < v_t < \dots < v_2 < v_1 \text{ and } t = \lfloor (s+1)/2 \rfloor.$$

We first consider the case $(1, n) \in P$. We can represent the order in H of the vertices of V_P and of the subintervals by the following string:

$$u_1 v_1 [i_{s-1}, j_{s-1}] [i_s, j_s] v_2 u_2 [i_1, j_1] [i_2, j_2] u_3 v_3 [i_{s-3}, j_{s-3}] [i_{s-2}, j_{s-2}] v_4 u_4 [i_3, j_3] [i_4, j_4] u_5 v_5 \dots$$

Of course, the vertices of the subintervals are permuted with H as they were in case (1).

From the ladder construction, there are four recognizable *types* of subintervals, two types on the left and two types on the right. While we could write down subscript formulas for each of the four types, for the cutwidth argument it is sufficient to consider the following four representatives of the four types: $[i_3, j_3]$, $[i_4, j_4]$, $[i_{s-3}, j_{s-3}]$ and $[i_{s-2}, j_{s-2}]$. The only edges that add to $\text{cw}(H_k)$ are edges incident to vertices in V_P that pass over the k th subinterval in H . The diagram in Figure 5.21 illustrates the *potential* for a vertex in V_P to have edges incident to some subinterval. For example, u_2 or v_2 might have one or more edges to subintervals $[i_1, j_1]$, $[i_s, j_s]$, $[i_2, j_2]$, and $[i_{s-1}, j_{s-1}]$. Since we are interested only in an upper bound on cutwidth, we ignore the possibility that the existence of some edge may preclude the existence of other edges.

We start with the type represented by subinterval $[i_3, j_3]$. An examination of the string for H together with Figure 5.21 reveals the potential for edges passing over $[x_3, y_3]$ from u_3, v_3, u_4, v_4, u_5

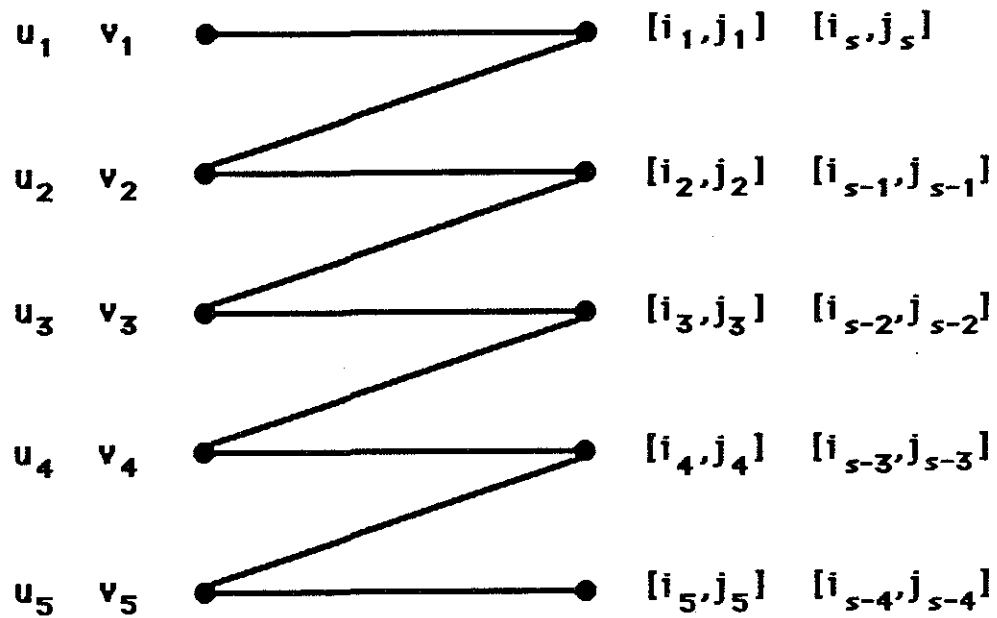


Figure 5.21. Proof of Lemma 5.4

and v_6 only. Hence, by inductive hypothesis,

$$\begin{aligned}
 \text{cw}([x_b, y_b]) &\leq 6d + \text{cw}(H_b) \\
 &\leq 6d + Cd \log(\text{size}\{[i_b, j_b]\}) \\
 &\leq 6d + Cd \log \frac{2}{3}n \\
 &\leq (Cd \log n) - 2d \\
 &< (Cd \log n) - \text{def}\{[1, n]\}
 \end{aligned}$$

since each subinterval contains at most $\frac{2}{3}n$ vertices.

Similarly, consideration of the three types represented by $[i_4, j_4]$, $[i_{s-3}, j_{s-3}]$ and $[i_{s-2}, j_{s-2}]$ reveals that at most 6 vertices in V_P can have incident edges adding to the cutwidth of a subinterval. Hence, for all subintervals $[x_b, y_b]$ in H ,

$$cw(\{x, y\}) \leq (Cd \log n) - \text{def}\{[1, n]\}.$$

Consideration of intervals in H between the subintervals (e.g., $[u_b, v_b]$) yields no worse an upper bound. Hence we conclude that $cw(H) \leq (Cd \log n) - \text{def}\{[1, n]\}$.

The case in which $(1, n) \notin P$ is similar to the preceding case. The additional left or right subinterval cannot boost the cutwidth above $(Cd \log n) - \text{def}\{[1, n]\}$. Hence in all cases, I.2 holds.

This completes the induction and the proof of the lemma. \square

5.5. Performance

In this section, we analyze the time and space complexity of TRADEOFF. Of course, the complexity depends on the representation of data. While we do not prescribe the details of the data representation, we do require that the representation make elementary operations efficient (i.e., constant time per edge or vertex). A place where this requirement is crucial is Algorithm 5.3 for finding a separating edge. To accomplish step 1 in linear time, it is necessary to be able to recognize the next (counterclockwise) edge of an interior face in constant time. It is easy to represent G so that this is possible.

First, we note that all operations of TRADEOFF performed on G except the recursive calls require linear time. From the description of the steps in section 5.3, all steps are clearly linear time except steps 6 and 9. From the description of the string construction, G' can be constructed in linear time from the $\{G_i\}$ (step 6). Similarly, the ladder construction can be accomplished in linear time (step 9). Hence, the entire algorithm excluding recursive calls can be implemented in linear time.

Let $T(n)$ be the time complexity of TRADEOFF when $n=n$. Let n_1, n_2, \dots, n_p be the sizes of the subintervals either in step 5 or in step 9, depending on which case holds. Then, $\sum_{i=1}^p n_i \leq n$ and $n_i \leq \frac{2}{3}n, 1 \leq i \leq p$. By the result of the previous paragraph, there exists a constant c such that

$$T(n) \leq cn + \sum_{i=1}^p T(n_i).$$

Lemma 5.5. If $T(1)$ is one unit of time, then for all $n > 1$,

$$T(n) \leq (c/\log \frac{2}{3})n \log n.$$

Proof: By induction on n . The lemma is certainly true for $n=2$. Assume $n > 2$ and assume the truth of the lemma for values smaller than n . Then,

$$\begin{aligned} T(n) &= cn + \sum_{i=1}^2 T(n_i) \\ &\leq cn + \sum_{i=1}^2 (c/\log \frac{3}{2})n_i \log n_i \\ &\leq cn + (c/\log \frac{3}{2}) \sum_{i=1}^2 n_i \log \frac{2}{3}n \\ &= cn + (c/\log \frac{3}{2})n \log \frac{2}{3}n \\ &= cn + (c/\log \frac{3}{2})n \log n - (c/\log \frac{3}{2})n \log \frac{3}{2} \\ &= (c/\log \frac{3}{2})n \log n. \end{aligned}$$

The lemma follows by induction. \square

The space requirements of TRADEOFF are clearly n times some small constant. We thus have the following:

Theorem 5.6. TRADEOFF has time complexity at most $C_1 n \log n$ and space complexity at most $C_2 n$, for small constants C_1, C_2 .

5.6. Conclusion

In this chapter, we have investigated tradeoffs between pagewidth and pagenumber that are significant in a VLSI context. Our main result is an algorithm for obtaining a book embedding for outerplanar graphs that is within a constant factor of optimal in VLSI area for the class of outerplanar graphs. While this near-optimality is not guaranteed for *individual* outerplanar graphs, we know of no example of an outerplanar graph for which our algorithm fails to obtain near-optimal area.

CHAPTER 6

CONCLUSIONS

We have presented three algorithms for embedding graphs in books. Each algorithm guarantees the quality of the book embedding that it generates. Hence the correctness of each algorithm constitutes a proof of a book embedding property for the class of graphs input to the algorithm. Each algorithm is efficient in time and space, hence of practical value for embedding graphs in books.

Our first algorithm embeds any planar graph in a book of at most seven pages. Thus we have shown that PPG , the pagewidth of the class of planar graphs, is at most seven, the smallest upper bound currently known.

Our second algorithm embeds any trivalent planar graph in a two-page book. In particular, our algorithm edge-augments any trivalent planar graph to obtain a planar hamiltonian graph in time linear in the size of the graph. Thus we have shown that MV , the maximum valence of a planar graph that guarantees the graph is subhamiltonian, is at least three.

Our third algorithm embeds any d -valent n -vertex outerplanar graph in a two-page book with at most $Cd \log n$ pagewidth, $C=8/(\log \frac{3}{2})$; the algorithm executes in time $O(n \log n)$. Moreover, we know of no outerplanar graph for which our algorithm fails to attain pagewidth within a small constant factor of the cutwidth of the graph. We show that at the cost of one additional page above optimal pagewidth, layouts of near-optimal cutwidth for outerplanar graphs can be obtained constructively.

Our results are applicable to the motivating VLSI problems in Chapter 1. For the multilayer VLSI layout problem, our results bound the number of layers required to realize circuits represented by planar graphs. Of particular interest is the outerplanar graph result; it bounds both the number of layers (i.e., two) and the area (essentially proportional to $n \log n$ for an n -component circuit). For the DIOGENES design problem, our results bound the number of stacks required to realize planar graphs. The outerplanar graph result also bounds the area of a DIOGENES layout for a circuit represented by an outerplanar graph.

Our algorithms are based on principles that should apply to the development of book embedding algorithms for other classes of graphs. It is of extreme importance to choose the order of vertices intelligently so as to make small pagewidth or small pagewidth possible. If an algorithm can find a cycle in the input graph whose removal separates the graph into two parts, it has the basis for a recursive decomposition. However, it is essential that the book embedding solution for each part preserve cycle order, or else the two solutions cannot be joined together to obtain a solution for the entire graph. In a more general approach, a separating subgraph other than a cycle can be chosen provided that it is possible to preserve the same order for the vertices of the separating subgraph in the solution of each subproblem.

To lend evidence that it will not be easy to extend our results to other classes of graphs, we present the following sequence. For $n=1,2, \dots$, G_n is a trivalent graph having $2n$ vertices. Its vertex set is $\{v_1, \dots, v_{2n}\}$, and its edge set is

$$\{(v_k, v_{k+1 \pmod{2n}}) | 1 \leq k \leq 2n\} \cup \{(v_k, v_{n+k}) | 1 \leq k \leq n\}.$$

Figure 6.1 illustrates G_4 . It is easy to show that G_n is a genus-one (i.e., embeddable on a torus) graph, but not a planar graph. There is an obvious hamiltonian cycle H for G_n : $(v_1, v_2, \dots, v_{2n})$. Using the order of H to obtain a book embedding for G_n requires an n -page book. However, G_n is just the n -ladder (Chapter 5) with added edges $(v_n, v_{n+1}), (v_{2n}, v_1)$. The ladder-like order

$$(v_1, v_{n+1}, v_{n+2}, v_2, v_3, v_{n+3}, \dots)$$

yields a three-page embedding for G_n . This example demonstrates that merely choosing a hamiltonian cycle in a (genus-one) graph does not guarantee a good book embedding. (The pinwheel in

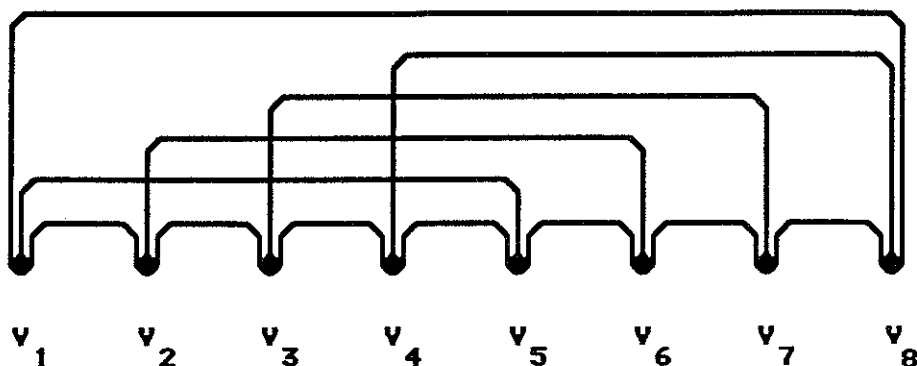


Figure 6.1. The Genus-One Graph G_4

[CLR] is another such example.)

We wish to suggest some directions for future research. It would be most interesting to determine PPG . We have investigated a sequence of graphs similar to the stellations of K_3 (Chapter 3) that we call *nested triangles*. The first nested triangle is $N_1=K_3$. The k -th nested triangle $N_k, k>1$ is derived from N_{k-1} by adding a triangle to each interior face of N_{k-1} and connecting edges from the triangle to the face. Figure 6.2 illustrates N_2 . We have attempted to obtain three-page embeddings for this sequence using the same approach that succeeded for the stellations of K_3 . The attempts failed. We conjecture that some member of this sequence requires at least four pages in any book embedding. The proof of this conjecture would imply that $PPG \geq 4$. We also recognize the existence of some freedom remaining in our planar graph algorithm. We believe that $PPG < 7$.

Another fruitful area for research is tradeoffs between pagenumber and pagewidth. It is not known how prevalent such tradeoffs are or whether dramatic tradeoffs exist for any pagenumber. In a VLSI context, algorithms for embedding a graph in a bounded number of pages with pagewidth close to the cutwidth of the graph could be most practical. We do not know of any example where adding one or two pages above the pagenumber of G does not give us an embed-

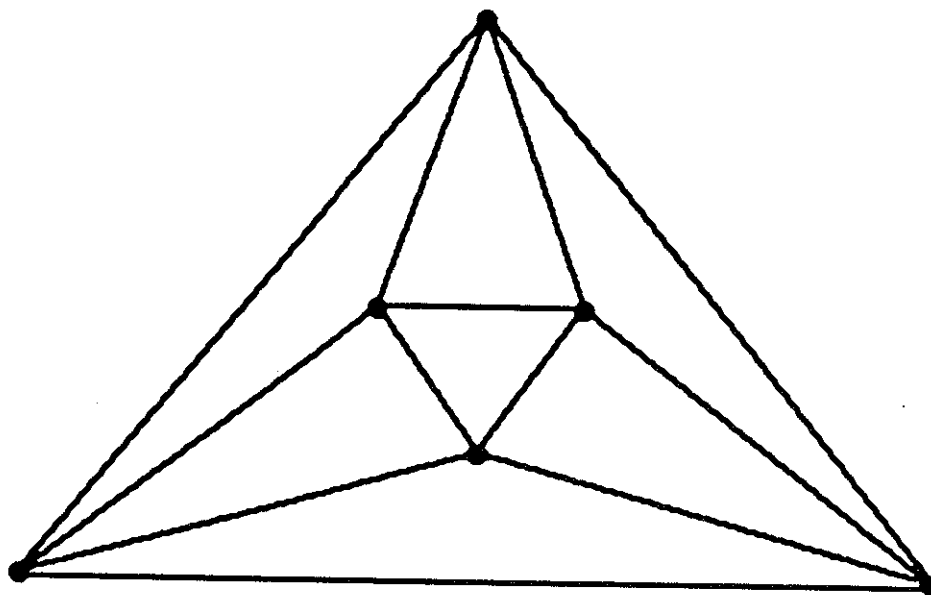


Figure 6.2. The Second Nested Triangle N_2

ding whose pagewidth is within a small constant factor of the pagewidth of G ; there is hope that such algorithms exist for important classes of graphs. In particular, we believe that such an algorithm is possible for planar graphs. The algorithm would embed any d -valent planar graph in a B -page book with $Cd\sqrt{n}$ pagewidth where C is a small constant. Our planar graph algorithm provides the starting point for obtaining bounded pagenumber. The small pagewidth will depend on a stronger version of Lipton and Tarjan's [LT] planar separator theorem.

REFERENCES

- [BK] F. Bernhart and P.C. Kainen (1979): The book thickness of a graph. *Journal of Combinatorial Theory (B)* 27, 320-331.
- [BB] A.J. Blodgett and D.R. Barbour (1982): Thermal conduction module: a high-performance multilayer ceramic package. *IBM Journal of Research and Development* 26, 30-36.
- [BS] J.F. Buss and P.W. Shor (1984): On the pagenumber of planar graphs. *16th ACM Symposium on Theory of Computing*, 98-100.
- [CM] M. Capobianco and J.C. Molluzzo (1978): *Examples and Counterexamples in Graph Theory*. Elsevier North-Holland, Inc., New York.
- [CLR] F.R.K. Chung, F.T. Leighton, A.L. Rosenberg (1984): Embedding graphs in books: A layout problem with applications to VLSI design. Submitted for publication. See also *5th International Conference on Theory and Applications of Graphs*.
- [Ev] S. Even (1979): *Graph Algorithms*. Computer Science Press, Rockville, Maryland.
- [EI] S. Even and A. Itai (1971): Queues, stacks, and graphs. In *Theory of Machines and Computations* (Z. Kohavi and A. Paz eds.) Academic Press, NY, pp. 71-86.
- [Ga] R.A. Games (1985): Optimal book embeddings of the baseline, Benes and barrel shifter networks. Submitted for publication.
- [GGJK] M.R. Garey, R.L. Graham, D.S. Johnson, D.E. Knuth (1978): Complexity results for bandwidth minimization. *SIAM Journal on Applied Mathematics* 34, 477-495.

- [GJMP] M.R. Garey, D.S. Johnson, G.L. Miller, C.H. Papadimitriou (1980): The complexity of coloring circular arcs and chords. *SIAM Journal on Algebraic and Discrete Methods* 1, 216-227.
- [Gav] F. Gavril (1977): Some NP-complete problems on graphs. *Proceedings of the Eleventh Conference on Information Sciences and Systems*, John Hopkins University, Baltimore, Maryland, 91-95.
- [Ha] F. Harary (1969): *Graph Theory*. Addison-Wesley Publishing Company, Reading, Massachusetts.
- [HP] F. Harary and Palmer (1966): The block-cutpoint-tree of a graph. *Publicationes Mathematicae Debrecen* 13, 103-107.
- [He] L.S. Heath (1984): Embedding planar graphs in seven pages. *25th IEEE Symp. on Foundations of Computer Science*, 74-83.
- [LR1] F.T. Leighton and A.L. Rosenberg (1983): Automatic generation of three-dimensional circuit layouts. *IEEE International Conference on Computer Design*, 633-636.
- [LR2] F.T. Leighton and A.L. Rosenberg (1984): Three-dimensional circuit layouts. *SIAM Journal on Computing*, to appear.
- [Le] C.E. Leiserson (1980): Area-efficient graph layouts (for VLSI). *21st IEEE Symp. on Foundations of Computer Science*, 270-281.
- [Len] T. Lengauer (1982): Upper and lower bounds on the complexity of the min-cut linear arrangement problem on trees. *SIAM Journal on Algebraic and Discrete Methods* 3, 99-113.
- [LT] R.J. Lipton and R.E. Tarjan (1979): A separator theorem for planar graphs. *SIAM Journal of Applied Mathematics* 36, 177-189.
- [Lo] W.R. Locke (1983): Three-dimensional integration: a critical survey. In *Microelectronics Center of North Carolina Technical Report 83-06*.

- [RS] R. Raghavan and S. Sahni (1983): Single row routing. *IEEE Transactions on Computers C-32*, 209-220.
- [Ro1] A.L. Rosenberg (1981): Three-dimensional integrated circuitry. In *VLSI Systems and Computations* (ed. H.T. Kung, B. Sproull, G. Steele) Computer Science Press, Rockville, MD, pp. 69-80.
- [Ro2] A.L. Rosenberg (1983): Three-dimensional VLSI: A case study. *Journal of the ACM 30*, 397-416.
- [Ro3] A.L. Rosenberg (1983): The DIOGENES approach to testable fault-tolerant arrays of processors. *IEEE Transactions on Computers, C-32*, 902-910.
- [RT] P. Rosenstiehl and R.E. Tarjan (1984): Gauss codes, planar hamiltonian graphs, and stack-sortable permutations. *Journal of Algorithms 5*, 375-390.
- [So] H.C. So (1974): Some theoretical results on the routing of multilayer printed-wiring boards. *1974 IEEE International Symposium on Circuits and Systems*, 296-303.
- [Sy] M.M. Syslo (1979): Characterizations of outerplanar graphs. *Discrete Mathematics 26*, 47-53.
- [Ta] R.E. Tarjan (1972): Sorting using networks of queues and stacks. *Journal of the ACM 19*, 341-346.
- [TK] B.S. Ting and E.S. Kuh (1978): An approach to the routing of multilayer printed circuit boards. *1978 IEEE International Symposium on Circuits and Systems*, 902-911.
- [TKS] B.S. Ting, E.S. Kuh, I. Shirakawa (1976): The multilayer routing problem: algorithms and necessary and sufficient conditions for the single-row single-layer case. *IEEE Transactions on Circuits and Systems, CAS-23*, 768-778.
- [Wh] H. Whitney (1931): A theorem on graphs. *Annals of Mathematics 32*, 378-390.
- [Wi] A. Wigderson (1982): The complexity of the hamiltonian circuit problem for maximal planar graphs. Princeton University EECS Department Report 298.

GLOSSARY

Adjacent: Two vertices are adjacent if there is an edge between them.

Two faces in a planar embeddings are adjacent if their boundaries have an edge in common.

Biconnected: A graph $G=(V,E)$ is biconnected if for every pair $u,v \in V$, there exists a cycle in G containing both u and v . This definition is equivalent to G not having a cutpoint.

Biconnected component: A maximal biconnected subgraph.

Bipartite graph: A graph $G=(V,E)$ is bipartite if V can be partitioned into subsets V_1 and V_2 such that every edge joins a vertex in V_1 to a vertex in V_2 .

Book: A line (the spine) together with some numbers of halfplanes (the pages) having the line as boundary.

Book embedding: A linear embedding of a graph in a book such that each edge of the graph is assigned to a single page of the book in such a way that on each page, the edges assigned to that page do not intersect.

Boundary: If F is a face in a planar embedding, then the boundary of F consists of those vertices and edges making up the topological boundary of the connected region of F .

Bounding cycle: If the boundary of a face F is a cycle (this is the case whenever the planar graph is biconnected), it is the bounding cycle of F .

Complete graph: A complete graph on n vertices, denoted K_n , has every pair of n vertices adjacent.

Connected: A graph $G=(V,E)$ is connected if for every pair $u,v \in V$, there exists a path in G from u to v .

Connected component: A maximal connected subgraph.

Cutpoint: A vertex whose removal increases the number of connected components.

Cutwidth: Given a graph G and a linear embedding of G , the cutwidth of a point, p , on the line is the number of edges having one endpoint on the left of p and the other on the right of p . The cutwidth of the linear embedding is the maximum cutwidth over all p .

Cycle: A cycle in a graph (multigraph) G is a list of three or more (one or more) distinct vertices (v_1, v_2, \dots, v_m) such that v_k is adjacent to v_{k+1} , $1 \leq k < m$, and v_m is adjacent to v_1 . m is the length of the cycle.

Degree: The degree of a vertex is the number of edges incident to it.

Depth-first search: A method of visiting each vertex of a graph exactly once.

Dual: The dual of a planar embedding of $G=(V,E)$ is a multigraph $G^D=(V^D,E^D)$ defined as follows: $V^D=\{F|F \text{ is a face}\}$, and $E^D=\{(F,F')|F \text{ is adjacent to } F'\}$.

Endpoint: If (u,v) is an edge, then u and v are its endpoints.

Exterior face: The unique unbounded face of a planar embedding.

Face: Given a planar embedding of a planar graph G , a face of the embedding is a maximal connected region in the complement of the planar embedding.

Genus: A surface has genus k if it is homeomorphic to a sphere with k attached handles. The genus of a graph G is the minimum k such that G can be embedded in a surface of genus k .

Hamiltonian: A graph is hamiltonian if it has a hamiltonian cycle.

Hamiltonian cycle: A cycle containing all vertices of the graph.

Incident: An edge and a vertex are incident if the vertex is an endpoint of the edge.

Interior face: Any bounded face of a planar embedding.

Linear embedding: Any embedding in which the vertices of a graph are ordered on a line. Book embedding is an example.

Loop: An edge incident to only one vertex, i.e., an edge of the form (u,u) .

Matching: A matching in a graph is a set of edges, no two of which have an endpoint in common.

Maximal planar graph: A graph to which no edge can be added without rendering it non-planar.

Multigraph: A generalization of the concept of graph where any edge may occur more than once, i.e., parallel edges are allowed.

MV: The maximum valence of a planar graph that guarantees the graph is subhamiltonian.

Outerplanar: A graph is outerplanar if it has a planar embedding in which all vertices are on the boundary of the exterior face.

Pagenumber: The pagenumber of a book embedding is the number of pages in the book.

The pagenumber of a graph G is the minimum pagenumber of any book embedding of G .

The pagenumber of a class of graphs is the minimum number of pages that every member of the class can be embedded in, as a function of graph size.

Pagewidth: The width of a page is the maximum number of edges that intersect any half-line perpendicular to the spine in the page. The pagewidth of a book embedding is the maximum width of any page.

The pagewidth of the graph G is the minimum pagewidth of any book embedding of G in a book having a minimum number of pages.

The pagewidth of a class of graphs is the minimum pagewidth that every member of the class can be embedded in, as a function of graph size.

Parallel edges: Edges that occur more than one time in a multigraph.

Path: A path in a graph G is a list of two or more distinct vertices (v_1, v_2, \dots, v_m) such that v_k is adjacent to v_{k+1} , $1 \leq k < m$. $m-1$ is the length of the path.

Planar embedding: A planar embedding of a graph G maps each vertex of G to a distinct point in the plane, and each edge (u,v) of G to a simple curve joining u and v , such that no two curves intersect.

Planar graph: A graph G is planar if it has a planar embedding.

PPG: The pagenumber of the class of planar graphs.

Quadrivalent: A quadrivalent graph has valence at most four.

Simple: A graph is simple if it contains no loops and no parallel edges.

Size: The size of a graph is the cardinality of its vertex set.

Subgraph: A graph $G'=(V',E')$ is a subgraph of a graph $G=(V,E)$ if $V'\subset V$ and $E'\subset E$.

Subhamiltonian: A graph is subhamiltonian if it is a subgraph of a planar hamiltonian graph.

Supercycle: A supercycle of a graph G is a cycle in a supergraph of G .

Supergraph: A graph $G'=(V',E')$ is a supergraph of a graph $G=(V,E)$ if $V=V'$ and $E\subset E'$.

Superhamiltonian cycle: A superhamiltonian cycle of a graph G is a hamiltonian cycle in a supergraph of G . The term is especially applied when the supergraph is planar.

Three-connected: A graph is three-connected if the removal of fewer than three vertices does not disconnect the graph.

Triangle: A face in a planar embedding whose boundary is a cycle of length three.

Triangulate: Add edges to a face of a planar embedding so that, in the resulting planar embedding, the face is partitioned into triangles.

Triangulated: A planar graph (or a planar embedding) is triangulated if all its faces are triangles. This is equivalent to the graph being a maximal planar graph.

Trivalent: A trivalent graph has valence at most three.

Valence: The valence of a graph is the maximum degree of any of its vertices.