Karlsruhe Institute of Technology

Book Embedding with Fixed Page Assignments

Bachelor Thesis of

Daniel Hoske

At the Department of Informatics
Institute of Theoretical Computer Science

Reviewers:	Prof. Dr. Dorothea Wagner
	Prof. Dr. Peter Sanders
Advisors:	Dipl.-Inform. Thomas Bläsius
	Dr. Ignaz Rutter

Time Period: June 2012 to September 2012

Statutory Declaration

I hereby declare that this thesis is the result of my own work, that I used no other than the indicated references and resources, that all the information that has been taken directly or indirectly from other sources is indicated as such, and that I have regarded the statute of the Karlsruhe Institute of Technology on securing good scientific practice in its currently applicable version.

Karlsruhe, August 8, 2013

Abstract

A k-*page book embedding* of a graph is a drawing of that graph in a book, with vertices along the book's *spine* (a straight line) and edges in k of the book's *pages* (half planes with the spine as boundary) such that the edges do not cross. In this thesis we consider the problem of determining whether such a drawing exists when the assignment of edges to pages is predetermined.

We start by showing that this problem is NP-complete for an unbounded number of pages, even if the edges on each page form a matching, and then solve some special cases. In the case of connected graphs on each page, we provide a linear-time decision algorithm. When the graphs on each page are disjoint perfect matchings, we show that the graph has to be bipartite to be embeddable and give bipartite examples and counterexamples.

Following these results, we consider several variations of the problem. Firstly, if we constrain the vertex orders on the spine by a PQ-tree only containing Q-nodes as inner nodes, embeddability can be decided in quadratic time. Secondly, we alter the embedding problem by taking multiple spines (parallel lines) in the plane and associating every vertex with a spine the vertex has to be drawn on. Additionally, edges must be drawn between consecutive spines, above the topmost spine or below the bottommost spine. We show that this variation is equivalent to a special case of the 2-page book embedding problem with fixed page assignments where the vertex order is constrained by a PQ-tree only containing P-nodes as inner nodes.

At the end we outline the most important open problems for book embedding with fixed page assignments and provide some suggestions on how to approach them.

Deutsche Zusammenfassung

Eine k-*Seiten Bucheinbettung* eines Graphens ist eine Zeichnung dieses Graphens in einem Buch mit Knoten auf dem *Buchrücken* (einer Geraden) und Kanten in k der *Seiten* des Buches (Halbebenen mit dem Buchrücken als Rand). Dabei dürfen sich Kanten nicht kreuzen. In dieser Arbeit betrachten wir das Problem, die Existenz einer solchen Zeichnung zu prüfen, wenn die Zuordnung von Kanten zu Seiten fest ist.

Wir beginnen damit, die NP-Vollständigkeit des Bucheinbettungsproblems für eine unbeschränkte Anzahl von Seiten zu zeigen. Das Problem bleibt selbst dann NP-vollständig, wenn die Graphen auf den Seiten Matchings sind. Danach lösen wir einige Spezialfälle des Problems.

Zunächst geben wir für den Fall zusammenhängender Graphen auf den Seiten einen Linearzeitalgorithmus an.

Danach fordern wir, dass die Graphen auf den Seiten disjunkte perfekte Matchings bilden. Einbettbare Graphen müssen in diesem Fall bipartit sein und wir konstruieren bipartite Beispiele und Gegenbeispiele.

Nach diesen Spezialfällen betrachten wir verschiedene Varianten des Bucheinbettungsproblems. Zuerst wird die Ordnung der Knoten auf dem Buchrücken mittels eines PQ-Baumes eingeschränkt, der nur Q-Knoten als innere Knoten hat. In diesem Fall ist Einbettbarkeit in quadratischer Zeit entscheidbar.

Anschließend betrachten wir mehrere Buchrücken (parallele Geraden) in der Ebene und ordnen jeden Knoten einem dieser Buchrücken zu, auf dem er gezeichnet werden muss. Des Weiteren sollen Kanten in dieser Variante zwischen den Buchrücken, oberhalb des obersten Buchrückens und unterhalb des untersten Buchrückens gezeichnet werden. Diese Variante ist äquivalent zu einem Spezialfall von 2-seitiger Bucheinbettung, wobei die Knotenordnung zusätzlich durch einen PQ-Baum beschränkt wird, der nur P-Knoten als innere Knoten hat.

Abschließend skizzieren wir die wichtigsten offenen Probleme für Bucheinbettungen mit fester Kantenzuordnung und bestimmen einige Ansatzpunkte für deren Lösung.

Acknowledgements

I would like to express my deepest gratitude to my advisors Dr. Ignaz Rutter and Dipl.-Inform. Thomas Bläsius for their continued support, guidance, advice as well as for the insightful discussions I had with them and to Prof. Dorothea Wagner for giving me the chance of writing this thesis at her institute. Special thanks also go to every other proofreader of this thesis and anyone else who supported me during its writing in any shape or form.

Contents

1. Introduction

Graphs play an important role in modelling many types of systems or relationships of data, such as communication networks, evolutionary trees in biology or the friend relationship in social networks. Moreover, graph drawings and especially planar embeddings are important for layouting transistors on a VLSI-chip. For large graphs even the human aptitude for solving visual problems no longer suffices for determining drawings in reasonable time, and an algorithmic approach on a computer has to be used.

A special drawing problem is k-*page book embedding*, which was first introduced by Ollmann in 1973 [**Ollmann73**]. It asks whether there is an embedding of a graph in a book with vertices along the book's *spine* (a straight line) and edges in k of the book's *pages* (half planes with the spine as boundary) such that the edges do not cross. The *book thickness* of a graph G is the smallest k such that G is k-page book embeddable.

One application for book embedding is the automatic placement of components and the wiring between them on a multilayer printed circuit board. A wire may cross layers but no two distinct wires may intersect in the same layer. The goal is then to find a good placement (we do not define what this actually means here).

An approach to this problem, first introduced by So [**So74**], is to arrange the components on a regular grid and group elements together that should appear in the same row or column. By introducing dummy elements, we* can make sure that

*Although this thesis has just one author, the first person plural is used instead of the singular. This should not be understood as pluralis maiestatis but as an invitation to the reader to follow the thought processes of the author and is quite usual in mathematical treatises.

wires connect elements in a single row or column. That is, we can layout the circuit board by layouting each row and column individually. The variant of this single-row situation—introduced by Raghavan and Sahni [**Raghavan83**]—that does not allow wires to either cross a row or to change layers then directly corresponds to a 2-page book embedding problem. This example was taken from a paper by Chung et. al. [**Chung87**] that also provides several other practical applications for book embedding, justifying the usefulness of the book embedding problem.

In this thesis we consider a more restricted version of the book embedding problem where each edge is assigned to the page is has to be embedded in, i. e. we fix the *page assignments*. In the remainder of this chapter we formally define the book embedding problem (Section 1.1), present related work (Section 1.2) and highlight the contribution of this thesis (Section 1.3).

1.1. The Book Embedding Problem

In this section we first present some basic definitions from graph theory. Then we formally formulate the problem BOOK-EMBEDDING that we consider in this thesis.

A *graph* G is a pair (V, E) where V is a non-empty finite set and $E \subseteq \binom{V}{2}$. The set V is called the set of *vertices of* G and E is the set of *edges of* G. We use $V(G)$ to refer to the vertices of G and $E(G)$ to refer to the edges of G. A planar embedding of a graph G is a drawing of G in $\mathbb{R}^2$ such that edges do not intersect except at common endpoints. Planar embeddings are revisited more formally at the start of Chapter 2.

A page embedding is a special planar embedding.

Definition 1.1. A *page embedding* of a graph $G = (V, E)$ is a planar embedding of G such that the vertices of G lie on the *real line* $\mathbb{R} \times \{0\}$ and every edge lies in the *upper half-plane* $\{(x, y) \in \mathbb{R}^2 \colon y > 0\}$ apart from the edge's endpoints.

Section 1.1 depicts a page embedding for C_3, the cycle on three vertices. By taking multiple page embeddings, we get a book embedding.

Definition 1.2. A *book embedding* of graphs $G_1 = (V, E_1), \ldots, G_k = (V, E_k)$ on the same set of vertices consists of page embeddings for each of the graphs that coincide in their vertex positions.

In the setting of book embeddings the line $\mathbb{R} \times \{0\}$ is also called the *spine*. We only demand that edges in the same graph G_i do not intersect. This can also be interpreted as giving each graph G_i its own upper half-plane, which we also call the *page of* G_i. Whenever we refer to a page in this thesis, we usually mean the graph G_i on this page.

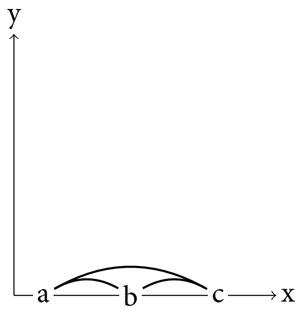


Figure 1.1.: A page embedding of C_3

The embeddability problem now asks whether such an embedding exists.

Problem: BOOK-EMBEDDING
Given: A vertex set V and edge sets $E_1, \dots, E_k \subseteq \binom{V}{2}$
Question: Is there a book embedding of $(V, E_1), \dots, (V, E_k)$?

In the literature *book embedding* usually refers to the somewhat different problem NOT-FIXED-BOOK-EMBEDDING. Instead of directly embedding k graphs into k pages, we first have to get k graphs by arbitrarily partitioning the edges of a graph into k parts and then embed these into k pages. The problem we call *book embedding* is often called *book embedding with fixed page assignment* in the literature.

Problem: NOT-FIXED-BOOK-EMBEDDING
Given: A vertex set V, an edge set $E \in \binom{V}{2}$ and a number $k \geqslant 1$
Question: Is there a partition $E = \bigcup_{i=1}^{k} E_i$ such that $(V, E_1), \dots, (V, E_k)$ is book embeddable?

We depict this assignment of the edges to the pages that has already been fixed either by using different colours or in the case of exactly two pages by one set of edges being drawn above and one below the spine.

1.2. Related Work

The BOOK-EMBEDDING problem is a graph drawing problem that is closely related to simultaneous planar drawings. Bläsius, Kobourov and Rutter [**GD2012**] provide a good overview of those types of problems.

In this section we first list some useful results about the usual book embedding problem NOT-FIXED-BOOK-EMBEDDING. Then we present the literature about its variant BOOK-EMBEDDING. The fixed page problem BOOK-EMBEDDING had not been

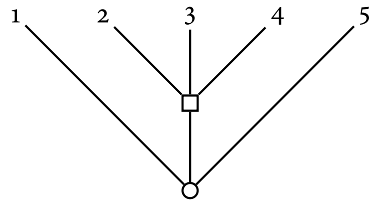


Figure 1.2.: A simple PQ-tree. P-nodes are drawn as a circle ○, Q-nodes as a box □.

well-studied before the writing of this thesis. Thus, the list of published results is quite small, even though it is exhaustive.

Furthermore, we introduce the concept of a PQ-tree here, as first described by Booth and Lueker [**Booth76**]. Although PQ-trees are not directly related to book embeddings, we use them in many of the special cases in Chapter 4. Thus, we also reference some results about PQ-trees at the end of the section.

Definition 1.3. A *PQ-tree* T on $M := \{1, \ldots, n\}$ is a rooted, ordered tree with leaves M and inner nodes either of type P or type Q.

The tree represents a set of permutations $\pi(T) \subseteq \mathrm{Sym}(M)$ on its leaves as follows: The order of the children of a P-node can be permuted in any way, while the order of the children of a Q-node can only be reversed. The set $\pi(T)$ then consists exactly of the permutations of the leaves M that we can get by flipping the inner nodes in any of the specified valid ways.

The empty set of permutations on a set cannot normally be described by a PQ-tree, so we define the special *null tree* ε representing $\emptyset$. Furthermore, let a *P-tree* be a PQ-tree containing only P-nodes as inner nodes and a *Q-tree* be a PQ-tree only containing Q-nodes as inner nodes.

For example, the PQ-tree in Figure 1.2 represents the permutations 12345, 14325, 52341, 54321, 15234, 15432, 51234, 51432, 23415, 43215, 23451 and 43251.

Book Embedding without Fixed Page Assignments

We first enumerate some results for NOT-FIXED-BOOK-EMBEDDING. We do not use these results, but they are still a helpful point of reference for classifying our findings about the fixed page case BOOK-EMBEDDING.

There are a variety of results about small page numbers: The graphs embeddable in a single page are just the outerplanar graphs and a graph is embeddable into two pages if and only if it is *sub-hamiltonian*, i. e. a subgraph of a planar Hamiltonian graph, as shown by Bernhart and Kainen [**Bernhart79**]. Widgerson [**Widgerson82**] proved that checking a maximal planar graph for Hamiltonicity is NP-complete, i. e. NOT-FIXED-BOOK-EMBEDDING is already NP-complete if we restrict ourselves to two pages.

Thus, the book embedding problem for $k \geqslant 2$ pages is probably not efficiently solvable. One viable research direction following this insight was to consider variations of the problem or special graphs. Indeed, there are results for several graphs, including but not limited to the following: A planar graph is embeddable into four pages [**Yannakakis86**], a graph of treewidth k has book thickness at most $k + 1$ [**Dujmovic07**] and genus g graphs have book thickness $\mathcal{O}(\sqrt{g})$ [**Malitz88**]. We do not explain these results further as we do not use them.

Book Embedding with Fixed Page Assignments

In contrast, little is known about the variant BOOK-EMBEDDING with fixed page assignments. The only result we could find appears in a recently published technical report by Hong and Nagamochi [**two-page-09**]. They show that BOOK-EMBEDDING is decidable in linear time for two pages, unlike NOT-FIXED-BOOK-EMBEDDING. That is, the argument for the NP-completeness of NOT-FIXED-BOOK-EMBEDDING cannot be adopted for fixed page assignments.

Furthermore, Angelini et. al. [**angelini11**] provide an interesting application for the BOOK-EMBEDDING problem. They consider the important special case CONNECTED-SEFE, whose complexity is unknown, of the simultaneous embedding problem SEFE and reduce it to a 2-page BOOK-EMBEDDING problem where the vertex order on the spine is additionally constrained by a P-tree.

Problem: SEFE
Given: Two graphs G_1 and G_2.
Question: Are there planar embeddings of G_1 and G_2 that coincide on $G_1 \cap G_2$?

Problem: CONNECTED-SEFE
Given: Two graphs G_1 and G_2 where $G_1 \cap G_2$ is connected.
Question: Are there planar embeddings of G_1 and G_2 that coincide on $G_1 \cap G_2$?

PQ-trees

The concept of a PQ-tree was originally described by Booth and Lueker [**Booth76**] to solve the consecutive ones problem: Given a 0-1-matrix, is there a permutation of its columns such that the 1's in every row appear consecutively? Let T be a PQ-tree and S some subset of its leaves. Then they showed that there is an operation reduce(T, S), computable in $\mathcal{O}(|S|)$ time, that yields a PQ-tree representing exactly the permutations in $\pi(T)$ where all leaves in S appear consecutively. Thus, the consecutive ones problem can be solved by starting with a PQ-tree representing all permutations on the columns (a single P-node) and reducing on the columns containing 1's row by row. This reduction operation proves useful for solving the variations on book embedding in Chapter 4.

Booth [**Booth75**] additionally showed that we can intersect two PQ-trees in linear time, i. e. from two PQ-trees T_1 and T_2 on the same leaves we can construct a PQ-tree representing $\pi(T_1) \cap \pi(T_2)$ in linear time.

We also make use of the fact that there are linear time planarity algorithms that test a graph G for planarity and embed G vertex-by-vertex. The general scheme for these algorithms is summarised by Haeupler and Tarjan [**Haeupler08**]. Their scheme unifies and simplifies several similar linear-time planarity algorithms:

1. The Lempel-Even-Cederbaum algorithm [**Lempel67**] that was refined to run in linear time by Booth and Lueker [**Booth76**]
2. The Shih-Hsu algorithm [**Shih93**]
3. The Boyer-Myrvold algorithm [**Boyer99**]

1.3. Contribution and Outline

From the literature analysis above we can see that there are a lot of open problems for BOOK-EMBEDDING. For example: Is BOOK-EMBEDDING NP-complete for a linear number of pages? Does BOOK-EMBEDDING remain efficiently solvable for 3 pages? What happens when we constrain the vertex orders by a PQ-tree?

In this thesis we want to solve some of these open problems and provide points of reference for further research. We first consider the time complexity of BOOK-EMBEDDING and then some special cases or restrictions thereof. More specifically, this work is structured as follows:

Chapter 2 Firstly, we provide basic definitions and results we need in the rest of the thesis. This is followed by rephrasing BOOK-EMBEDDING as a total ordering problem ORDER-BOOK-EMBEDDING. At the end of the chapter we begin to actually study book embeddings by identifying some of the freedoms we have in choosing total orders that solve an ORDER-BOOK-EMBEDDING instance.

Chapter 3 As alluded to above, in this chapter we show that the BOOK-EMBEDDING problem is NP-complete for an unbounded number of pages by reduction from the betweenness problem that is also defined in this chapter. This result even applies if the edges on each page form a matching.

Since we still want to decide book embeddability for some graphs, we then show how to solve BOOK-EMBEDDING in exponential time by expressing book embeddability using 3-CNF-formulae. We also provide some optimisations for these formulae.

Chapter 4 This chapter contains our main results. We discuss and solve BOOK-EMBEDDING for several special cases and restrictions.

Firstly, we show that BOOK-EMBEDDING can be solved in linear time if each page is a connected graph on all vertices.

Then we proceed with the opposite, the almost completely disconnected case, i. e. we take disjoint perfect matchings as pages. We already know that BOOK-EMBEDDING is NP-complete for (general) matchings on the pages, at least if we allow an unbounded number of pages. It is not clear whether perfect matchings make BOOK-EMBEDDING easier.

Still, we can give a necessary criterion when the pages are perfect matchings, namely the union of all the pages has to be a bipartite graph. This criterion is not sufficient. Indeed, we then find both bipartite examples and counterexamples for all numbers of pages, partly using computer assistance.

We continue with a variation on book embedding. Motivated by a result of Angelini et. al. [**angelini11**], we consider book embeddings with the additional constraint that the order of the vertices must be represented by a given Q-tree. We show that this restricted case is solvable in quadratic time.

The chapter closes with another variation on BOOK-EMBEDDING. We take multiple spines (parallel lines) in the plane and associate every vertex with a spine it has to be drawn on. Additionally, we impose the restriction that edges must be drawn between consecutive spines, above the topmost spine or below the bottommost spine. Unlike the previous case, we do not solve this variation. We just show that it is equivalent to BOOK-EMBEDDING with 2 pages where the vertex order is constrained by a special kind of P-tree. If we did not allow the edges to go above the topmost or below the bottommost spine, this variation would be the same as the level planarity problem, which was first introduced by Tomii et. al. [**Tomii77**]. Jünger, Leipert and Mutzel presented an algorithm that checks for level planarity in linear time [**Junger99**].

Chapter 5 The thesis is concluded by summarising the results we obtained and discussing viable future research directions.

2. Preliminaries

As we noted in the introduction, this thesis is concerned with book embeddings with fixed page assignments, a variation on the problem of embedding graphs in the plane without edge crossings.

Let us first review some basic results from graph theory before beginning to deal with book embeddings. For a more detailed treatment of the basics, consult any introductory text on graph theory, e. g. the book "Graph Theory" by Diestel [**Diestel**].

Basic definitions

Often, we want to draw graphs in the Euclidean plane, identified with $\mathbb{R}^2$, without edge crossings. Such a drawing is called a *planar embedding*. If G is embeddable in the plane, we call it a *planar graph*.

More formally, a *planar embedding* of a graph $G = (V, E)$ consists of two maps m_V from V to $\mathbb{R}^2$ and m_E from E to the set of continuous functions from $[0, 1]$ to $\mathbb{R}^2$ with the following properties:

1. The endpoints of the curve corresponding to an edge are the images of the curve's incident vertices under m_V, i. e. $m_E(e)(0) = m_V(a)$ and $m_E(e)(1) = m_V(b)$ for all $e = \{a, b\} \in E$.
2. The curve $m_E(e)$ is injective (a *Jordan curve*) and piecewise differentiable for all $e \in E$.
3. Let $e_1, e_2 \in E$ be two different edges of G. Then $m_E(e_1)$ and $m_E(e_2)$ do not intersect in their interiors $m_E(e_1)\big((0, 1)\big)$ and $m_E(e_2)\big((0, 1)\big)$.

4. No image $m_E(e)$ of an edge $e \in E$ contains the image of any vertex apart from its endpoints.

Removing the images $m_E(e)$ of the edges e separates the plane into several connected components. These components are called the *faces* of the embedding.

The graphs that can be drawn planarly such that all vertices are on the boundary of a single face are the *outerplanar graphs*, a subset of the set of planar graphs. Let G be a graph and $\widetilde{G}$ be the extension that we get from G by adding a new vertex adjacent to all vertices of G. Clearly, G is outerplanar if and only if $\widetilde{G}$ is planar.

Another basic fact is Euler's formula. It states that any planar embedding of a graph with v vertices and e edges contains exactly $f = 2 + e - v$ faces. An edge bounds at most two faces and a face's boundary contains at least three edges, i. e. $3f \leqslant 2e$ (exception: a single edge). With Euler's formula we can, thus, bound the number of edges of a planar graph by $6 + 3e - 3v = 3f \leqslant 2e \iff e \leqslant 3v - 6$. Applying this bound to the extended planar graph belonging to an outerplanar graph yields $e + v \leqslant 3(v + 1) - 6 \iff e \leqslant 2v - 3$ since the extended graph has v additional edges and 1 additional vertex. Thus, an n-vertex outerplanar graph has at most $2n - 3$ edges. Equality occurs for C_n, the cycle on n vertices, with additional edges from one vertex to all of the $n-3$ other vertices it is not adjacent to (a triangulation of C_n).

Book embedding

With these basic definitions out of the way, we can return to the page embeddings of Definition 1.1. The problem of deciding whether a page embedding exists is not very interesting. The embeddable graphs are exactly the outerplanar graphs [**Bernhart79**]. Thus, we get the necessary condition $|E| \leqslant 2|V| - 3$ for page embeddability.

The problem becomes significantly harder when we have k pages as for the book embeddability problem BOOK-EMBEDDING. For the case $k = 1$ we just saw that checking for book embeddability is the same as checking for outerplanarity, which can be done by adding one vertex adjacent to all other vertices and testing planarity. Hong and Nagamochi [**two-page-09**] showed that BOOK-EMBEDDING is solvable in linear time for $k = 2$. In particular, it remains efficiently solvable. The general problem for arbitrary k had not been considered before this thesis.

We already know that $|E_i| \leqslant 2|V| - 3$ for all $i \in \{1, \ldots, k\}$ is necessary for book embeddability. That is, the number of edges $|E_i|$ is linear in $|V|$ for all $i \in \{1, \ldots, k\}$. Furthermore, in all complexity considerations we assume the number of pages to be constant. All in all, the size of a book embedding instance is in $\mathcal{O}(|V|)$.

We now have some idea of the problem we want to consider. To familiarise ourselves even more with book embeddings, firstly, this chapter reduces BOOK-EMBEDDING to a total ordering problem ORDER-BOOK-EMBEDDING that we can work with more easily (Section 2.1). Secondly, we show what freedoms we have in choosing total orders that solve a ORDER-BOOK-EMBEDDING instance (Section 2.2). This allows us to gain insight into the choices the problem leaves us, which proves useful for showing NP-completeness in Section 3.1.

2.1. Book Embedding as Total Ordering Problem

At first sight the book embedding problem looks like a geometric problem, where the actual page drawings are important. Since there are lots of different embeddings that differ just slightly in how they map the edges to curves, this would make the problem quite unwieldy.

In this section we show that the first impression deceives. Only the order of the vertices on the spine is significant for testing embeddability. Thus, we can turn the book embedding problem into a total ordering problem. In this thesis we always mean strict total orders when we speak about total orders.

We first show that a single page embedding corresponds to an ordering problem.

Definition 2.1. Let (V, E) be a graph and $<$ a total order on V. We call the condition that the suborder $a < c < b < d$ does not occur for any $\{a, b\}, \{c, d\} \in E$ the *book constraint for* E. If $<$ fulfils the book constraint for E we say that it is a *valid book order for* E.

Lemma 2.2. *There is a page embedding for* $G = (V, E)$ *if and only if there is a valid book order* $<$ *for* E.

Proof.

"$\Rightarrow$" Let there be a page embedding of G with vertex map m_V and edge map m_E. We use m_V to define the ordering via $u < v :\Leftrightarrow m_V(u) < m_V(v)$ for all $u, v \in V$.

If $a < c < b < d$ occurs for some $\{a, b\}, \{c, d\} \in E$, then the Jordan curves $m_E(\{a, b\})$—from a to b—and $m_E(\{c, d\})$—from c to d—have to intersect, as illustrated in Figure 2.1, since they lie in the upper half-plane.

More formally, we can add the spine to any page embedding by drawing straight lines between consecutive vertices on the spine and a curve in the upper half-plane from the leftmost vertex on the spine to the rightmost vertex on the spine such that planarity is preserved. This construction results in an outerplanar embedding since all vertices still lie in the component the *lower half-plane* $\{(x, y) \in \mathbb{R}^2 : y < 0\}$ belongs to. By doing this construction for the given page embedding of the edges $\{a, b\}$ and $\{c, d\}$ we get an outerplanar embedding of the complete graph on four vertices K_4, as illustrated in Figure 2.1. This is a impossible since K_4 is not outerplanar. Thus, the book constraint must be fulfilled.

"$\Leftarrow$" Let $<$ be valid book order for E and $i(v)$ denote the index of v in $<$ for all $v \in V$, i. e. $i(v) = j$ if and only if v is the j-th smallest element in V according to $<$.

Then we define the page embedding as follows. The vertex map m_V maps a vertex v to the real number $i(v)$ and the edge map m_E maps an edge $\{a, b\}$ to the semi-circle in the upper half-plane that has the line segment $m_V(a)m_V(b)$ as diameter.

Now let $e_1 := \{a, b\}, e_2 := \{c, d\}$ be different edges of G. If two of a, b, c and d are the same, the semi-circles $m_E(e_1)$ and $m_E(e_2)$ of different size share an endpoint, i. e. they do not intersect in their interior and we are done.

So assume a, b, c and d are pairwise distinct as well as

$$a < b, c < d \text{ and } a < c \tag{1}$$

without loss of generality.

The curve $m_E(e_1)$ is a semi-circle with centre $\big(i(a) + i(b)\big)/2$ and radius $\big(i(b) - i(a)\big)/2$. Similarly, the curve $m_E(e_2)$ is a semi-circle with centre $\big(i(c) + i(d)\big)/2$ and radius $\big(i(d) - i(c)\big)/2$. Since we assumed (1), these semi-circles intersect in their interior if and only if $i(c) < i(b) < i(d)$. With $a < c$ this would mean $a < c < b < d$ in contradiction to the assumption.

Therefore, no two semi-circles corresponding to different edges intersect in their interior, i. e. (m_V, m_E) is a valid page embedding. $\square$

Since the pages in a book embedding problem are independent of each other, we can use this lemma to rephrase the book embedding problem as a total ordering problem.

Problem: **ORDER-BOOK-EMBEDDING**
Given: A finite set $V := \{1, \ldots, n\}$ and sets $E_1, \ldots, E_k \subseteq \binom{M}{2}$.
Question: Is there a valid book order $<$ for all E_i where $i \in \{1, \ldots, k\}$?

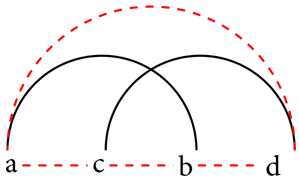
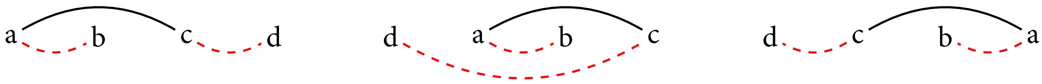


Figure 2.1.: If $a < c < b < d$ occurs for two edges $\{a, b\}$ and $\{c, d\}$, there is no page embedding with the order $<$. Especially not a canonical one using semi-circles. The red (dashed) edges result from adding the spine to the page embedding.



(a) (b) (c)

Figure 2.2.: A valid book embedding (a) with cyclic shift to the right by one element (b) and mirror image (c).

Theorem 2.3. *BOOK-EMBEDDING and ORDER-BOOK-EMBEDDING are equivalent.*

Proof. Follows by applying Lemma 2.2 to each page. □

From this point onward, we use both representations of the book embedding problem interchangeably. Whenever we refer to BOOK-EMBEDDING we also mean ORDER-BOOK-EMBEDDING and vice versa. Note that the book constraints for two edges $\{a, b\}$ and $\{c, d\}$ are trivially fulfilled if the edges have a common vertex. Therefore, we always assume that two edges are independent whenever we check book constraints in the remainder of the thesis.

2.2. Equivalent Orders

We significantly reduced the number of basically equivalent ways to solve a book embedding instance by restating the drawing problem BOOK-EMBEDDING as an ordering problem ORDER-BOOK-EMBEDDING. Still, several choices remain. We determine some of them in this section, namely that the mirror image and any cyclic shift of a total order solving a BOOK-EMBEDDING instance still solve the instance.

Notably, this knowledge proves useful for reducing other problems to book embedding in Chapter 3 where we discuss the time complexity of BOOK-EMBEDDING.

By considering all mirror images and cyclic shifts of a total order to be the same we turn the total order into what we call a *symmetric order*.

Let M be an n-element set and O_M be the set of total orders (permutations) on M.

For a permutation $\pi = (a_1, a_2, \ldots, a_n) \in O_M$ we say that $(a_n, a_{n-1}, \ldots, a_1) \in O_M$ is the *mirror image of* π and $(a_{n-k+1}, a_{n-k+2}, \ldots, a_n, a_1, \ldots, a_{n-k}) \in O_M$ is the *cyclic shift of* π *by* k for $k \in \{0, \ldots, n-1\}$. Both of these elementary operations are illustrated in Figure 2.2.

We now define a relation $\sim$ on O_M. For all $a, b \in O_M$ we have $a \sim b$ if and only if we can get b from a by a series of cyclic shifts and mirror images. Clearly, the relation $\sim$ is an equivalence relation.

Let $\widetilde{O}_M$ be the set of equivalence classes of O_M with respect to $\sim$. An element $o \in \widetilde{O}_M$ is then called a *symmetric order on* M. We write $[\pi] := \{\tau \in O_M : \pi \sim \tau\} \in \widetilde{O}_M$ to refer to the symmetric order corresponding to a permutation $\pi \in O_M$.

Let $<$ be a total order (permutation) on a finite set M. We can order the elements of M on a circle by starting at one point of the circle, going either clockwise or counter-clockwise without making a full turn and writing the elements of M on distinct points in the order $<$.

Conversely, if we have an arrangement of a finite number of elements on distinct points of a circle, it is ambiguous which total order we got this arrangement from. We can cut the circle open between any two elements and get a total order by going either clockwise or counter-clockwise. Clearly, the notion of a symmetric order is defined exactly in such a manner that the orders we get from $<$ (written on a circle) by cutting the circle open are the orders $[<]$.

That is, a symmetric order can alternatively be interpreted as a way of ordering elements on a circle. If we only consider cyclic shifts in the definition of $\sim$, we get *cyclic orders* which are considered more often in the literature.

We now show that cyclic shifts and mirror images preserve the validity of an order.

Theorem 2.4. *Let* $(V, E_1), \ldots, (V, E_k)$ *be a book embedding instance with valid book order* $< \in O_V$. *Then any order* $<_c \in [<]$ *is valid.*

Proof. We show that the cyclic shift by one $<_1$ of $<$ and the mirror image $<_2$ of $<$ are still valid orders. Then any order $<_c \in [<]$ must also be valid.

To do so we show that a page (V, E_i) cannot have the forbidden substructure of Lemma 2.2 in either order $<_1$ or $<_2$.

If $a <_2 c <_2 < b <_2 d$ occurs for some $\{a, b\}, \{c, d\} \in E_i$, we have $d < b < c < a \in E_i$ as $<_2$ is the reverse of $<$, which contradicts the validity of $<$. Thus, $<_2$ must be valid.

Similarly, assume the forbidden substructure $a <_1 c <_1 b <_1 d$ occurs for some $\{a, b\}, \{c, d\} \in E_i$. If a is not the smallest element of V with respect to $<_1$, we get $a < c < b < d$ by construction of $<_1$, a contradiction to the validity of $<$. Otherwise, we get $c < b < d < a$, which again contradicts the validity of $<$. Thus, $<_1$ is valid.

3. Algorithmic Complexity

The book embedding problem without fixed page assignments NOT-FIXED-BOOK-EMBEDDING is NP-complete. For two pages Bernhart [**Bernhart79**] showed that the problem is the same as determining whether the graph is *sub-hamiltonian*, i. e. a subgraph of a planar graph with a Hamiltonian cycle. This implies that NOT-FIXED-BOOK-EMBEDDING is NP-complete by a result of Widgerson's [**Widgerson82**], which states that the Hamiltonian circuit problem for maximal planar graphs is NP-complete.

Since the two page case with fixed partitions is solvable in linear time [**two-page-09**], we see that fixing the partitions significantly changes the book embedding problem. Is BOOK-EMBEDDING even NP-complete? We answer this question in the affirmative in the first half of this chapter (Section 3.1), but, unfortunately, only for an unbounded number of pages.

Thus, we know that BOOK-EMBEDDING is probably not efficiently solvable. In spite of that, we want to test some specific instances in Section 4.2. For this reason, in the second half of this chapter (Section 3.2) we show how BOOK-EMBEDDING can be solved in super-polynomial time by reducing it to 3-SAT with some optimisations. We chose the 3-SAT problem since there are solvers for it that work well on instances occurring in practice, even though 3-SAT is NP-complete.

3.1. Book-Embedding is NP-Complete

In this section we construct a polynomial-time reduction from the NP-complete problem BETWEENNESS, defined below, to BOOK-EMBEDDING, i. e. we show that

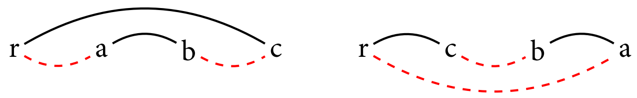


Figure 3.1.: The two drawings of a C_4 starting at r

BOOK-EMBEDDING is NP-complete. Therefore, we cannot expect there to be an efficient algorithm for solving the general book embedding problem.

Checking the book constraints for a pair of edges takes $\mathcal{O}(1)$ time and there are $\mathcal{O}(\sum_i |E_i|^2)$ pairs to check. Thus, checking the validity of a guessed book order takes polynomial time and the book embedding problem must, therefore, be in NP.

We now give a polynomial time reduction from the problem BETWEENNESS, which was shown to be NP-complete by Opatrny [**opatrny:79**], to BOOK-EMBEDDING and, thereby, show that BOOK-EMBEDDING itself is NP-complete.

Problem: BETWEENNESS
Given: A finite set $M := \{1, \dots, n\}$ and a set of ordered triples $C \subseteq M^3$.
Question: Is there a total ordering $<$ of M such that either $a < b < c$ or $a > b > c$ occurs for all $(a, b, c) \in C$?

The idea of the reduction is to map each triple to edges on two new pages that form a C_4, a cycle on 4 vertices. By the following lemma, these two pages exactly represent the betweenness constraint.

Lemma 3.1. *Let* $V := \{r, a, b, c\}$, $E_1 := \{\{r, a\}, \{b, c\}\}$ *and* $E_2 := \{\{a, b\}, \{r, c\}\}$. *Then* $r < a < b < c$ *and* $r < c < b < a$, *depicted in Figure 3.1, are the only valid book embeddings of* E_1 *and* E_2 *with* r *as first vertex.*

Proof. Let $<$ be a valid book order on V. By Lemma 2.2 the two pages yield the constraints $r < b < a \Leftrightarrow r < c < a$ and $r < a < c \Leftrightarrow r < b < c$. Since r is the smallest element of $<$, this is the same as $b < a \Leftrightarrow c < a \Leftrightarrow c < b$, i. e. only the two cases $r < a < b < c$ and $r < c < b < a$ remain.

Now let's do the formal reduction.

Theorem 3.2. *There is a polynomial time reduction from* BETWEENNESS *to* BOOK-EMBEDDING. *Thus,* BOOK-EMBEDDING *is NP-complete.*

Proof. Let $I := (M, C)$ be a betweenness instance.

Construct a book embedding instance $f(I) := \left(V, \bigcup_{t \in C} (E_{t,1} \cup E_{t,2})\right)$ as follows. Take $V := M \cup \{r\}$ as vertex set where $r \notin M$ is a new symbol. For each triple

$\tau = (a, b, c) \in C$ introduce two new pages $E_{\tau,1} := \{\{r, a\}, \{b, c\}\}$ and $E_{\tau,2} := \{\{a, b\}, \{r, d\}\}$. The instance $f(I)$ can obviously be computed in polynomial time.

Now show that I is a positive instance if and only $f(I)$ is one.

"$\Rightarrow$" Let I be a positive instance of BETWEENNESS with valid total order $<$. Extend $<$ to V via $r < k$ for all $k \in M$. For each triple $(a, b, c) \in C$ we have $r < a < b < c$ or $r < c < b < a$, i. e. $<$ yields a valid embedding of the pages by Lemma 3.1. Thus, $<$ is a correct solution of $f(I)$.

"$\Leftarrow$" Let $f(I)$ be a positive instance of BOOK-EMBEDDING with valid total order $<$. By Theorem 2.4 we can assume without loss of generality that $<$ is rotated such that r is its smallest element. Then we have the situation of Lemma 3.1 for each triple $\tau = (a, b, c) \in C$ with the pages $E_{\tau,1}$ and $E_{\tau,2}$, i. e. $a < b < c$ or $c < b < a$. Therefore, the order $<$ restricted to M is indeed a valid solution of I. □

We conclude that BOOK-EMBEDDING is NP-complete. The reduction of Theorem 3.2 gives us even more. The pages it creates are matchings, i. e. even the special case MATCHINGS-BOOK-EMBEDDING of book embedding where the edges on each page form a matching remains NP-complete.

Problem: MATCHINGS-BOOK-EMBEDDING
Given: A vertex set V and matchings $E_1, \ldots, E_k \subseteq \binom{V}{2}$
Question: Is there a book embedding of $(V, E_1), \ldots, (V, E_k)$?

We do not know how complex the problem is when the edges on the pages form perfect matchings. This special case is considered in more detail in Section 4.2.

3.2. Reduction to 3-SAT

In the previous section we saw that BOOK-EMBEDDING is NP-complete. This implies that we cannot expect to solve a general book embedding instance in polynomial time. But we still want to be able to check some instances, e. g. to find counterexamples in special cases as in Section 4.2. We, therefore, make it the goal of this section to give a super-polynomial time algorithm for deciding book embeddability.

In order to achieve this, we express BOOK-EMBEDDING as a satisfiability problem of a Boolean formula. The translation immediately yields a Boolean formula in 3-CNF. That is, the formula is a conjunction of disjunctions of literals (positive or negative variables) and each disjunction contains at most 3 literals. The problem of deciding satisfiability for these Boolean formulae is called 3-SAT and has been studied extensively.

Problem: 3-SAT
Given: A 3-CNF Boolean formula f.
Question: Is f satisfiable?

Although 3-SAT was the first problem to be proven NP-complete [**Cook71**], there are SAT-solvers that handle many instances occurring in practice in reasonable times. One that has scored well in several contests is `minisat` [**minisat03**]. We use it to check the resulting formulae for some instances in Section 4.2.

Now we derive the translation of the total order formulation of BOOK-EMBEDDING from Lemma 2.2 into a Boolean formula. Let $\big((V, E_1), \dots, (V, E_k)\big)$ be the book embedding instance and label the vertices $V = \{1, \dots, n\}$ without loss of generality.

We have to express the total order $<$ as a set of Boolean variables. It is natural to introduce a variable $v(i, j)$ for the statement $i < j$ for all $i, j \in V$.

Then we have to build a 3-CNF formula that rephrases the stipulation that $<$ is a strict total order and that the book constraints are fulfilled. We can achieve this by forming the conjunction of the following formulae.

Strict total order

That $<$ is a strict total order means, by definition, that it is asymmetric, irreflexive, transitive and total:

irreflexive For each vertex $i \in V$ the formula $i < i$ is false. We get $\neg v(i, i)$. (n clauses)

asymmetric and total For each unordered pair of distinct vertices $i, j \in V$ exactly one of $i < j$ or $j < i$ is true. We get $v(i, j) \dot\vee v(j, i) \equiv \big(v(i, j) \vee v(j, i)\big) \wedge \big(\neg v(i, j) \vee \neg v(j, i)\big)$. (two clauses for each of the $\binom{n}{2}$ unordered pairs)

transitive For each triple $i, j, k \in V$ of vertices, if $i < j$ and $j < k$ are true, then also $i < k$. We get $\big(v(i, j) \wedge v(j, k)\big) \Rightarrow v(i, k) \equiv \neg v(i, j) \vee \neg v(j, k) \vee v(i, k)$. (one clause for each of the $n(n-1)(n-2)$ ordered triples of distinct vertices)

Book constraints

For each unordered pair of different edges $e_1 := \{a, b\}, e_2 := \{c, d\} \in E_i$, we have to take the book constraint from Definition 2.1 into account. The constraint says exactly that c is between a and b if and only if d is between a and b as well as that a is between c and d if and only if b is between c and d.

That is, if we choose one of the edges e_1 or e_2 as edge $e_O := \{w, x\}$ and the other as edge $e_I := \{y, z\}$, we get the following equivalence:

$$\big(v(w,y) \wedge v(y,x) \Leftrightarrow v(w,z) \wedge v(z,x)\big) \tag{1}$$

Exchanging the vertices y and z does not change the resulting clauses, while exchanging the vertices w and x does. Thus, there are two choices to make. Firstly, which of e_1 and e_2 gets the name e_O and which gets the name e_I (two possibilities). Secondly, which incident vertex of the edge e_O gets the name w and which gets the name x (two possibilities).

Since the formula (1) is equivalent to the CNF-formula $\big(\neg v(w,y) \vee \neg v(y,x) \vee v(w,z)\big) \wedge \big(\neg v(w,y) \vee \neg v(y,x) \vee v(z,x)\big) \wedge \big(\neg v(w,z) \vee \neg v(z,x) \vee v(w,y)\big) \wedge \big(\neg v(w,z) \vee \neg v(z,x) \vee v(y,x)\big)$, we get $2 \cdot 2 \cdot 4 = 16$ clauses for each of the $\sum_{i=1}^{k} \binom{|E_i|}{2}$ unordered pairs of edges.

We actually do not need the clauses for both choices of e_O. Once the SAT-formulae for one choice have been added, the constraints for the other choice immediately follow.

We now show this observation. Let $e_1 := \{a, b\}$, $e_2 := \{c, d\}$ be edges and assume we have the SAT-formulae of type (1) with $e_O = e_1$ as well as the SAT-formulae for the asymmetry and totality. We now show $v(c,a) \wedge v(a,d) \Rightarrow v(c,b) \wedge v(b,d)$. The other instances of (1) with $e_O = e_2$ can be proven analogously.

Assume that $v(c,a) \wedge v(a,d)$ is true. If $v(d,b)$ is true, we can infer $v(a,c)$ by $v(a,d) \wedge v(d,b) \Leftrightarrow v(a,c) \wedge v(c,b)$ (the formula (1) with $e_O = \{a,b\}$, $w = a$ and $x = b$) which contradicts $v(c,a)$ because of the asymmetry constraint. That is, $v(b,d)$ must be true by the asymmetry and totality. In the same manner we can show $v(c,b)$. Thus, the assumption implies $v(c,b) \wedge v(b,d)$, as desired.

This small optimisation saves half of the clauses, i. e. we only need eight clauses for each pair of edges.

Fixed minimum

From Theorem 2.4 we know that cyclic shifts preserve the validity of $<$. To help the SAT-solver, we can, therefore, assume that 1 is the smallest vertex and add the clauses $v(1,j)$ for all $j \in V$ with $j \neq 1$. They comprise another $n-1$ clauses.

Axiom	**3-CNF formula**	**Number of clauses**
Irreflexive	$\neg v(i,i)$	n
Asymmetric and total	$\big(v(i,j) \lor v(j,i)\big) \land \big(\neg v(i,j) \lor \neg v(j,i)\big)$ for all distinct unordered pairs $i,j \in V$	$2\binom{n}{2}$
Transitive	$\neg v(i,j) \lor \neg v(j,k) \lor v(i,k)$ for all ordered triples $i,j,k \in V$ where i,j and k are pairwise distinct	$n(n-1)(n-2)$
Book embedding	$\big(\neg v(w,y) \lor \neg v(y,x) \lor v(w,z)\big)$ $\land\big(\neg v(w,y) \lor \neg v(y,x) \lor v(z,x)\big)$ $\land\big(\neg v(w,z) \lor \neg v(z,x) \lor v(w,y)\big)$ $\land\big(\neg v(w,z) \lor \neg v(z,x) \lor v(y,x)\big)$ for all distinct $e_O = \{w,x\}$, $e_I = \{y,z\} \in E_i$ and all $i \in \{1,\dots,k\}$ where it matters which edge is assigned to e_O and which to e_I	$16 \cdot \sum_{i=1}^{k} \binom{\lvert E_i\rvert}{2}$
Book embedding (optimised)	$\big(\neg v(w,y) \lor \neg v(y,x) \lor v(w,z)\big)$ $\land\big(\neg v(w,y) \lor \neg v(y,x) \lor v(z,x)\big)$ $\land\big(\neg v(w,z) \lor \neg v(z,x) \lor v(w,y)\big)$ $\land\big(\neg v(w,z) \lor \neg v(z,x) \lor v(y,x)\big)$ for all distinct $e_O = \{w,x\}$, $e_I = \{y,z\} \in E_i$ and all $i \in \{1,\dots,k\}$ where it does not matter which edge is assigned to e_O and which to e_I	$8 \cdot \sum_{i=1}^{k} \binom{\lvert E_i\rvert}{2}$
Fixed minimum	$v(1,j)$ for all $j \in \big(V \setminus \{1\}\big)$	$n-1$

Table 3.1.: The 3-CNF formulae corresponding to BOOK-EMBEDDING

The clauses we get are summarised in Table 3.1. They provide us with a polynomial-time reduction from BOOK-EMBEDDING to 3-SAT. The number of edges $|E_i|$ is linear in $|V|$ for all $i \in \{1,\dots,k\}$ since (V, E_i) is an outerplanar graph. Thus, we get $\mathcal{O}\big(|V|^3 + k|V|^2\big)$ clauses.

Theorem 3.3. *Let* $I := (V, E_1, \dots, E_k)$ *be a* BOOK-EMBEDDING *instance. There is a* $\mathcal{O}\big(|V|^3 + k|V|^2\big)$ *time reduction to an equivalent* 3-SAT *instance with* $\mathcal{O}\big(|V|^3 + k|V|^2\big)$ *clauses.*

4. Special Cases and Restrictions

Having proven that BOOK-EMBEDDING is a hard problem in the previous section, we now turn to some special instances or variations on BOOK-EMBEDDING and either show how they can be solved efficiently or why they remain hard. This section is the main contribution of this thesis.

One topic we are interested in is how embeddability depends on the connectivity of the pages. Thus, we deal with both extreme cases with regards to connectivity in the first two sections: The pages can either all be connected (Section 4.1), or "maximally disconnected" without isolated vertices (Section 4.2), i. e. consisting of perfect disjoint matchings.

We show that the connected case—quite surprisingly—admits a solution in linear time. For each page we get a PQ-tree (see Definition 1.3) that stands for the possible orders of the vertices in a page embedding of this single page. The solution intersects these sets of orders to decide embeddability.

In contrast, we are unable to provide an efficient algorithm if the edges of each page form perfect disjoint matchings. We only manage to prove for this case that an embeddable graph must be bipartite. Furthermore, we derive—by hand and using a computer—positive bipartite instances and the smallest bipartite counterexamples for all numbers of pages with the exception of three pages. For three pages we are able to get a smallest counterexample when two of the matchings form a cycle.

Another interesting restriction we make in Section 4.3 is to allow the order of the vertices on the spine to only come from the permutations represented by a fixed PQ-tree. We present a quadratic time algorithm for solving the book embedding problem with this restriction if we only allow Q-trees. Angelini et. al. [**angelini11**] showed that a similar restriction is important by reducing CONNECTED-SEFE (see page 5) to a book embedding problem with 2 pages constrained by a P-tree.

Finally, in Section 4.4 we give a related variation on the book embedding problem. We now allow multiple spines (parallel lines) in the plane and associate every vertex with a spine the vertex has to be drawn on. Additionally, we demand that edges are drawn between consecutive spines, above the topmost spine or below the bottommost spine. We show that this variation is equivalent to a special case of the 2-page book embedding problem where the vertex order is constrained by a P-tree, i. e. it is indeed related to the problem of Section 4.3.

While the results we present are mostly independent of each other and small, they still provide significant insight into the book embedding problem and Section 4.3 is a step towards solving SEFE.

4.1. Connected Graphs

One approach for solving the book embedding problem is to determine the set of valid total orders (permutations) for each of the pages and obtain the valid book orders by intersecting these sets. Since there are $n!$ permutations on n vertices, this method is not efficient and even needs super-polynomial space. Indeed, we have shown in the previous chapter that we cannot expect there to be an efficient algorithm.

We know that cyclic shifts and mirror images do not matter. Considering this, we can encode the possible valid orders more efficiently by only storing one symmetric order in place of $2n$ total orders. However, this is still not sufficient for efficiently solving the book embedding problem. Are there better encodings that make the algorithm feasible, at least for special graphs?

As a matter of fact, there are. In this section we see that the valid total orders can be encoded very efficiently using PQ-trees if the pages are connected—the first special case of BOOK-EMBEDDING we consider. Furthermore, PQ-trees on the same vertices can also be efficiently *intersected*, i. e. we can efficiently get a PQ-tree representing $\pi(T_1) \cap \pi(T_2)$ from two PQ-trees T_1 and T_2 on the same leaves.

Problem: **CONNECTED-BOOK-EMBEDDING**
Given: A vertex set V and edge sets $E_1, \dots, E_k \subseteq \binom{V}{2}$ such that the graphs (V, E_i) are connected for $i \in \{1, \dots, k\}$.
Question: Is there a book embedding of $(V, E_1), \dots, (V, E_k)$?

Planarity testing using PQ-trees

There are several planarity testing algorithms that represent the possible planar embeddings using PQ-trees. A high-level scheme for these methods is described by Haeupler and Tarjan [**Haeupler08**]. It can be adapted to give all valid orders of a page embedding as we show below.

We first briefly describe this scheme. It embeds the graph G vertex by vertex. At each step we store the set of possible partial planar embeddings where some subset of the vertices has already been embedded.

The edges that have exactly one embedded endpoint at a step are *half-embedded*. If the non-embedded vertices form a connected graph at each step, the half-embedded edges must lie on a common face that we can without loss of generality assume to be the outer face, i. e. all the already embedded vertices incident to half-embedded edges are on the boundary of the outer face.

That the non-embedded vertices form a connected graph can be guaranteed by choosing a leaf-to-root order in any fixed spanning tree of the connected graph G. The possible partial embeddings can then be represented by the order of their half-embedded edges around the outside of their component.

It can be shown that these orders are given by a PQ-tree for every component of the subgraph induced by the embedded vertices. For a more detailed explanation consult the paper by Haeupler and Tarjan [**Haeupler08**]. They also show how to implement the scheme in linear time.

Representing book embeddings using PQ-trees

This is not yet what we want. A page embedding is an outerplanar embedding and not a planar embedding. Thus, we have to modify the planarity algorithm slightly. We build the connected graph $\widetilde{G}$ by adding a new vertex r to G that is adjacent to every vertex in $V(G)$. In Chapter 2 we noted the fact that G is outerplanar if and only if $\widetilde{G}$ is planar. Furthermore, by removing r from a planar embedding of $\widetilde{G}$ we get an outerplanar embedding of G.

Now we choose r as last vertex in the leaf-to-root order of the planarity algorithm. Since G is connected, the scheme above yields a single PQ-tree T representing all extendible planar embeddings of G (as possible orders of the half-embedded edges $\{r, v\}$ for $v \in V(G)$) at its penultimate step. By the argument above, every such embedding is outerplanar. Also, T not only gives the orders of the half-embedded edges but all vertices of G since r is adjacent to every other vertex, i. e. every vertex of G is the endpoint of exactly one half-embedded edge.

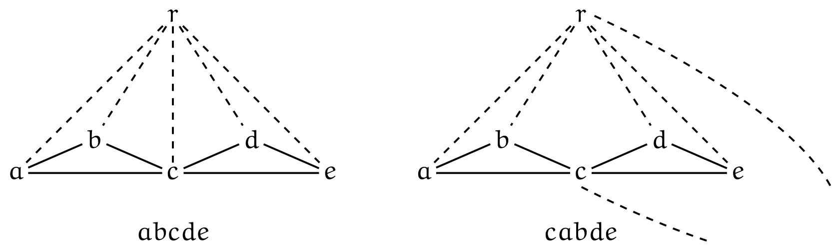


Figure 4.1.: The outerplanar embedding represents the vertex orders in both [$abcde$] and [$cabde$]. The symmetric orders [$abcde$] and [$cabde$] correspond to different edge orders around r in the extended graph.

A subtle distinction is still noteworthy, although it does not present any problems for the algorithm. When walking around the outer boundary of an outerplanar embedding in clockwise or counter-clockwise direction, we can meet a vertex twice. Thus, the outerplanar embedding in Figure 4.1, for example, can yield the total orders in both [$abcde$] and [$cabde$]; yet these orders belong to different planar embeddings of the extended graph.

In conclusion, there is a PQ-tree from which we can read all valid outerplanar embeddings (page embeddings). This PQ-tree can be computed in linear time as shown by Haeupler and Tarjan [**Haeupler08**].

Lemma 4.1. *Let $G = (V, E)$ be a connected graph. Then we can compute a PQ-tree representing all valid orders of the vertices V in a page embedding of G in $\mathcal{O}(|V|)$ time.*

To get the set of valid book orders, all that remains to be done is to intersect the PQ-trees we get. Say we want to intersect the PQ-trees S and T on the same leaves. Let v be an inner node of S and e one of its incident edges going to a child w of v. Then the leaves $C(w)$ that have w as ancestor appear consecutively in any order $\pi \in \pi(S)$. Additionally, if v is a Q-node and e' is a consecutive edge of e going from v to w', then the leaves $C(w) \cup C(w')$ also appear consecutively in any $\pi \in \pi(S)$. On the other hand, any order fulfilling these constraints is in $\pi(S)$. That is, we can get a tree representing $\pi(S) \cap \pi(T)$ by applying the reductions just described to the tree T. A trivial implementation of this approach would need a quadratic number of reductions, but Booth described in his Ph. D. thesis [**Booth75**] how to reduce the cost of intersection to linear time.

Now that we are able to intersect PQ-trees, we can summarise the linear-time solution of CONNECTED-BOOK-EMBEDDING.

Theorem 4.2. *CONNECTED-BOOK-EMBEDDING can be solved in linear time.*

Proof. Let $(V, E_1, \dots, E_k)$ be the CONNECTED-BOOK-EMBEDDING instance. First construct the k PQ-trees $T_1, \dots, T_k$ representing all valid page embeddings of the corresponding graphs $(V, E_1), \dots, (V, E_k)$, each in $\mathcal{O}(|V|)$. Then consecutively intersect T_1 with $T_2, \dots, T_k$ using time $\mathcal{O}((k-1)|V|)$, yielding the PQ-tree T representing all valid solutions of the instance. The instance possesses a solution if and only if $T \neq \varepsilon$, which can be decided in constant time. All in all, we need $\mathcal{O}(k|V|)$ time.

Outlook

When the graphs on the pages are not connected, we also get PQ-trees for the valid orders of each of their components. That is, we have a set of PQ-trees and must decide whether they possess a common order in order to solve the book embedding problem. The hurdle is that the trees do not need to have the same leaves.

Bläsius and Rutter [**Blasius11**] considered a more general variant of this PQ-tree intersection problem, called SIMULTANEOUS-PQ-ORDERING. They showed the NP-completeness of SIMULTANEOUS-PQ-ORDERING for an unbounded number of trees. Investigating restrictions of SIMULTANEOUS-PQ-ORDERING may help us deal with the book embedding problem, but we are not sure how.

4.2. Perfect Matchings

A diametrically opposed simplification to Section 4.1 is to take disjoint perfect matchings as graphs on the pages since matchings are—in a sense—the most disconnected graphs apart from isolated vertices. That is, we least expect to be able to adapt the result for connected graphs to this new setting. Note that this is only possible for an even number of vertices and remember that we have shown the NP-completeness when taking (not necessarily perfect) matchings as pages in Section 3.1.

Problem: PERFECT-MATCHINGS-BOOK-EMBEDDING
Given: Disjoint perfect matchings $E_1, \dots, E_k$ on a vertex set V.
Question: Is there a book embedding of $(V, E_1), \dots, (V, E_k)$?

In this section we first show that an embeddable instance of PERFECT-MATCHINGS-BOOK-EMBEDDING has to be *bipartite*, i. e. the *union graph* $(V, E_1 \cup \dots \cup E_k)$ must be bipartite. Then we prove that the problem does not become trivial for any number k of pages by providing positive and negative bipartite instances. For all k we get a partition of $K_{k,k}$, the complete bipartite graph with k left and k right vertices, as smallest (vertex-minimal) positive bipartite instance and for $k > 3$ we get another

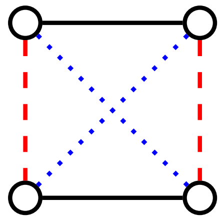


Figure 4.2.: A non-embeddable partition of K_4.

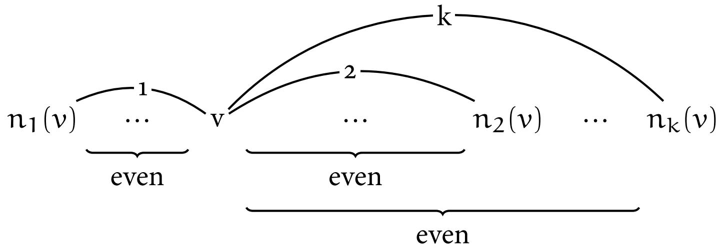


Figure 4.3.: There is an even number of vertices between v and each neighbour $n_i(v)$.

partition of $K_{k,k}$ as smallest negative bipartite instance. We show this by hand in Section 4.2.1. Getting a smallest bipartite counterexample for $k = 3$ is significantly more difficult and we have to resort to using computer assistance in Section 4.2.2. Even with the computer we only manage to find a smallest bipartite counterexample when two of the matchings are required to form a cycle.

The partition of K_4, the complete graph on four vertices, depicted in Figure 4.2 is already a counterexample to PERFECT-MATCHINGS-BOOK-EMBEDDING with three pages. This can be checked by hand or by using the corresponding 3-SAT instance we derived in Section 3.2. We observe that the main reason for its non-embeddability is the non-bipartiteness of K_4 since bipartiteness is necessary for embeddability, as we prove in the following theorem.

Theorem 4.3. *Let* $I := (V, E_1, \dots, E_k)$ *be an instance of* PERFECT-MATCHINGS-BOOK-EMBEDDING*. If the graph* $G := (V, E_1 \cup \dots \cup E_k)$ *is not bipartite, it has no book embedding.*

Proof. Assume we have a valid book order $<$ for G and let $s(v)$ be the index of $v \in V$, i. e. $s(v) = j$ if and only if v is the j-th smallest element in $<$. We show that G is bipartite with the bipartitions given by the parity of the index of a vertex, i. e. $V_E := \{v \in V : s(v) \text{ is even}\}$ and $V_O := V \setminus V_E = \{v \in V : s(v) \text{ is odd}\}$ is a bipartition of G.

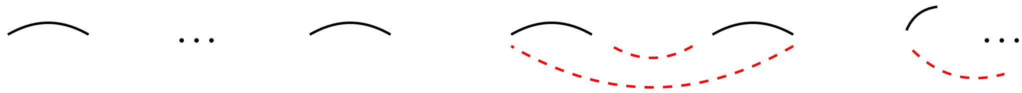


Figure 4.4.: One matching just consists of independent edges, while two matchings form disjoint cycles.

Let $i \in \{1, \ldots, k\}$. Every vertex $v \in V$ is incident to exactly one edge in each set E_i. We can, therefore, define $n_i(v)$ to be the unique neighbour of v in the graph (V, E_i). The neighbour $n_i(w)$ of a vertex w between $n_i(v)$ and v in the order $<$ has to occur between $n_i(v)$ and v since $<$ fulfils the book constraints. That is, the vertices between $n_i(v)$ and v appear in pairs. There is, therefore, an even number of vertices between $n_i(v)$ and v, as depicted in Figure 4.3, and the index of $n_i(v)$ has a different parity than the index of v.

We conclude that (V_O, V_E) is indeed a bipartition of G and bipartiteness is, therefore, necessary for book embeddability.

4.2.1. Bipartite Examples with at Least Four Pages

We know that non-bipartite graphs are not embeddable. But are there also bipartite counterexamples for all number of pages, or is the problem the same as testing bipartiteness? (That would be surprising since the slightly more general problem MATCHINGS-BOOK-EMBEDDING is NP-complete for a linear number of pages.)

At the other extreme, we ask whether there are positive instances for all number of pages. The number of edges in the whole graph is $(nk)/2$ for n vertices and k pages. That is, approximately k/n of the possible edges are present. Therefore, the resulting graph cannot be too small and it is not immediately apparent that there are positive bipartite instances for large k.

For one matching, G is a perfect matching which is obviously embeddable. For two matchings, every vertex of G has degree 2, i. e. G consists of disjoint even cycles. Thus, G is embeddable by placing the vertices of the cycles consecutively. These cases are illustrated in Figure 4.4.

We now consider the larger cases. Our goal is to provide both positive and negative bipartite instances for PERFECT-MATCHINGS-BOOK-EMBEDDING and all $k \geqslant 3$.

In this subsection we give examples for all $k \geqslant 4$ and prove their embeddability or non-embeddability, respectively, by hand. We believe that these proofs are vastly

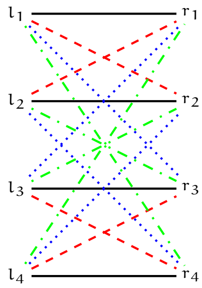


Figure 4.5.: A non-embeddable partition of $K_{4,4}$.

more illuminating than just using a SAT solver as a black-box. For the significantly more difficult case $k = 3$ we use computer assistance in the following subsection.

Surprisingly, the smallest bipartite graph $K_{k,k}$ that can be split into k perfect matchings provides us with both a positive and a negative instance; but we still have to choose the perfect matchings sensibly.

Consider the partition of $K_{4,4}$ of Figure 4.5. It was chosen such that any two matchings form two cycles of length 4. We prove that this partition is not embeddable.

Lemma 4.4. *The partition of* $K_{4,4}$ *given in Figure 4.5 is not book embeddable.*

Proof. We are looking for a valid book order $<$. From the proof of Theorem 4.3 we get that the left and right vertices have positions of different parity under $<$. Let v_i be the i-th smallest vertex under $<$ for $i \in \{1, \dots, 8\}$. Then v_1 is adjacent to v_2 and v_2 is adjacent to v_3 since our graph is $K_{4,4}$.

For reasons of symmetry, we can, therefore, assume $v_1 = l_1, v_2 = r_1$ and $v_3 = l_2$. By Lemma 3.1 this fixes the order of the vertices of both the black/blue (solid/dotted) C_4 containing l_1 and the black/green (solid/dash-dotted) C_4 containing r_1, i. e. $l_1 < r_1 < l_3 < r_3$ and $l_1 < r_1 < l_4 < r_4$. Since the left and the right vertices alternate, the black/red (solid/dashed) C_4 formed by $\{l_3, r_3, l_4, r_4\}$ now yields $l_3 < r_3 < l_4 < r_4$ or $l_4 < r_4 < l_3 < r_3$. Assume $l_3 < r_3 < l_4 < r_4$ in the following. The other case can be handled analogously.

The partial embedding we have so far is depicted in Figure 4.6. We see that the blue (dotted) edge $l_2 r_4$ intersects the blue (dotted) edge $r_1 l_3$. Thus, the graph is not book embeddable.

We can build partitions of $K_{k,k}$ into disjoint perfect matchings that contain the non-embeddable partition of $K_{4,4}$. These partitions of $K_{k,k}$ are then obviously also not embeddable.

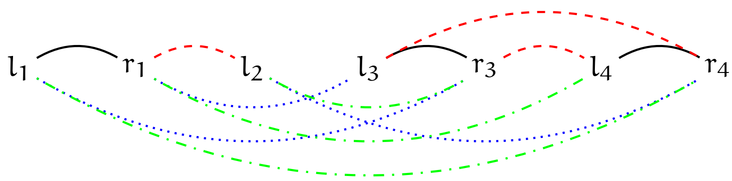


Figure 4.6.: Partial embedding of the partition in Figure 4.5.

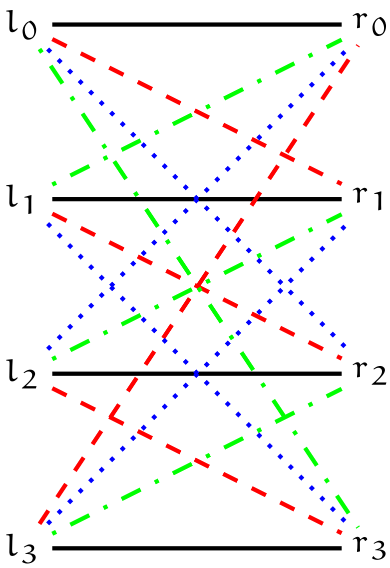


Figure 4.7.: The cyclic partition of $K_{4,4}$ is embeddable.

The positive instance we are looking for is somewhat harder to find since a graph containing a positive instance does not itself have to be book embeddable. That is, we have to explicitly give an embedding for every $k \geqslant 4$ and cannot just prove that some graph with $k = 4$ is embeddable and extend it to a partition of $K_{k,k}$ for larger k.

We label the left vertices with $\{l_0, \dots, l_{k-1}\}$ and the right vertices with $\{r_0, \dots, r_{k-1}\}$. It then turns out that the *cyclic partition* $E_i := \{\{l_j, r_{(j+i) \bmod k}\} : j \in \{1, \dots, k-1\}\}$ into matchings is embeddable. The case $k = 4$ is illustrated in Figure 4.7.

Lemma 4.5. *The complete bipartite graph $K_{k,k}$ with the cyclic partition is book embeddable.*

Proof. We get a valid order of the vertices by alternatingly listing the right vertices in increasing order and the left vertices in decreasing:

$$r_0 < l_{k-1} < r_1 < l_{k-2} < \cdots < r_{k-1} < l_0 \tag{1}$$

In the first matching E_0 the first vertex r_0 is matched with the last vertex l_0, the second vertex l_{k-1} with the penultimate vertex r_{k-1}, and so on. That is, the book constraints of Lemma 2.2 are fulfilled for the page E_0 in the order (1). More specifically, we get concentric semi-circles as canonical embedding in the proof of Lemma 2.2.

Now let $i \in \{1, \ldots, k-1\}$ and consider the matching E_i. Both E_i and E_0 are perfect matchings on the vertices of $K_{k,k}$, i. e. they are isomorphic. Still, E_i is somewhat harder to understand since it is shifted. To simplify the matching E_i we now want to relabel the right vertices r_j such that E_i matches l_j to r_j for each $j \in \{0, \ldots, k-1\}$. We achieve this by renaming $r_i, \ldots, r_{k-1}$ to $r_0, \ldots, r_{k-i-1}$ as well as $r_0, \ldots, r_{i-1}$ to $r_{k-i}, \ldots, r_{k-1}$ in this order.

After applying this relabelling to the original order, we get

$$r_{k-i} < l_{k-1} < \cdots < r_{k-1} < l_{k-i} < r_0 < l_{k-i-1} < \cdots < r_{k-i-1} < l_0. \tag{2}$$

Since E_i now matches each l_j to r_j, we see that when we cut the vertices between l_{k-i} and r_0, we get two independent sets of concentric semi-circles as canonical embedding (Lemma 2.2) of E_i in the order (2) which the same as the order (1) after a change of name. Thus, the book constraints are fulfilled for the page E_i. □

4.2.2. Bipartite Counterexamples with Three Matchings

As alluded to above, in this subsection we determine a smallest bipartite counterexample for three disjoint perfect matchings when two of the matchings form a cycle.

We first give the case with three pages a name.

Problem: 3-PERFECT-MATCHINGS-BOOK-EMBEDDING
Given: Three disjoint perfect matchings E_1, E_2 and E_3 on a vertex set V.
Question: Is there a book embedding of (V, E_1), (V, E_2), (V, E_3)?

The smallest possible 3-PERFECT-MATCHINGS-BOOK-EMBEDDING instance K_4 is already a non-bipartite counterexample, as shown at the start of this section. In contrast, we do not immediately see a bipartite counterexample. In fact, the smallest bipartite counterexample has at least 20 vertices, as we see below. It can, therefore, not be viably found without computer assistance. In this subsection we describe how we used the computer to do so.

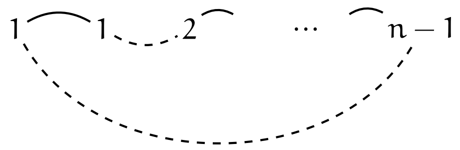


Figure 4.8.: Two matchings form a cycle.

We already know from Section 3.2 how a single book embedding instance can be tested using a SAT-solver. To look for a counterexample, we, naturally, just iterate over all bipartite instances of PERFECT-MATCHINGS-BOOK-EMBEDDING in increasing (even) order and test them for book embeddability.

This has to be done somewhat intelligently, using the symmetries of the problem, to remain in reasonable time. One improvement we use is to utilise multiple cores by letting the instance generator and the SAT-solvers run in parallel. How the solver stage can be accelerated by optimising the SAT-formulae was already discussed in Section 3.2. Below we show how to optimise the actual generator.

Even so, it is still too slow to get the smallest bipartite counterexample with the available computing hardware (4×12-Core AMD Opteron 6172, 2.1 GHz, 256 GB RAM). Therefore, we first provide the smallest counterexample for an even more restricted instance, namely that two of the matchings form a single cycle. It has order 28. We then proceed with the general 3-PERFECT-MATCHINGS-BOOK-EMBEDDING problem and compute that there is no bipartite counterexample with $\leqslant$ 18 vertices.

Two matchings form a cycle

At the start, we restrict ourselves to instances where two of the matchings form a cycle. Without loss of generality the cycle contains the vertices from 0 to $n-1$ in order, the first matching is $\big\{\{l,(l+1) \bmod n\}\colon l \in \{1,\dots,n-1\}, l \text{ even}\big\}$ and the second is $\big\{\{l,(l+1) \bmod n\}\colon l \in \{1,\dots,n-1\}, l \text{ odd}\big\}$, as depicted in Figure 4.8. This labelling already fixes the bipartition. The odd vertices form the first partition and the even vertices the second. The third matching can then be filled using backtracking by successively adding edges that do not already exist between vertices of different parity.

But there is another symmetry we can use, namely the rotational symmetry from Theorem 2.4. For this reason, we first define a value for edges that is invariant under cyclic shifts and can be interpreted as edge length in the corresponding symmetric order.

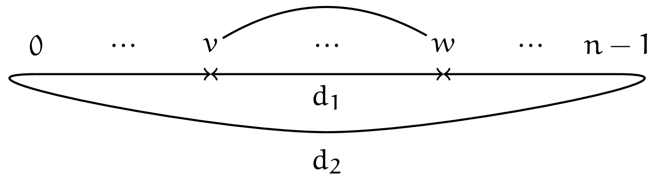


Figure 4.9.: The value $\mu(v) = \min\{d_1, d_2\}$ is the length of the edge $\{v, w\}$ in the symmetric order.

Definition + Lemma 4.6. *Let $<$ be a total order on $V := \{0, \ldots, n-1\}$, E a matching and $v \in V$. Furthermore, let w be the unique neighbour of v in E and $i : V \to V$ the index function of $<$.*

Then define $\mu(v) := \min\big\{|i(v) - i(w)|, n - |i(v) - i(w)|\big\}$. The value $\mu(v)$ is invariant under cyclic shifts of $<$.

Proof. If we consider the symmetric order $[<]$ corresponding to $<$, $\mu(v)$ can be interpreted as the length of the edge $\{v, w\}$ as in Figure 4.9. It is then clearly invariant under cyclic shifts.

We can, therefore, always rotate an instance such that the edge incident to 0 in the third matching has the largest length $\mu(\cdot)$. That is, we can first determine the edge incident to 0 in the backtracking process and need only consider edges with length at most $\mu(0)$ in the following backtracking steps.

Our implementation of this search strategy yields the graph in Figure 4.10 as one of the smallest counterexamples. In this example both the red/blue (dashed/dotted) pages and the red/black (dashed/solid) pages form cycles. There are other non-isomorphic bipartite counterexamples of this size that we do not depict.

Thus, a bipartite counterexample has at least 28 vertices in this special case. It may be possible to infer a useful sufficient condition for 3-PERFECT-MATCHINGS-BOOK-EMBEDDING from it. But we were unable to do so since the depicted graph is quite large and asymmetric.

No restrictions

If we abandon the restriction that two of the matchings form a cycle, we can proceed similarly. Without loss of generality the first matching connects every even vertex to the following odd vertex. Then the odd and even vertices form the bipartition. The remaining two matchings can then be filled in by backtracking and adding edges between vertices with different parity. By exchanging the second and third matching

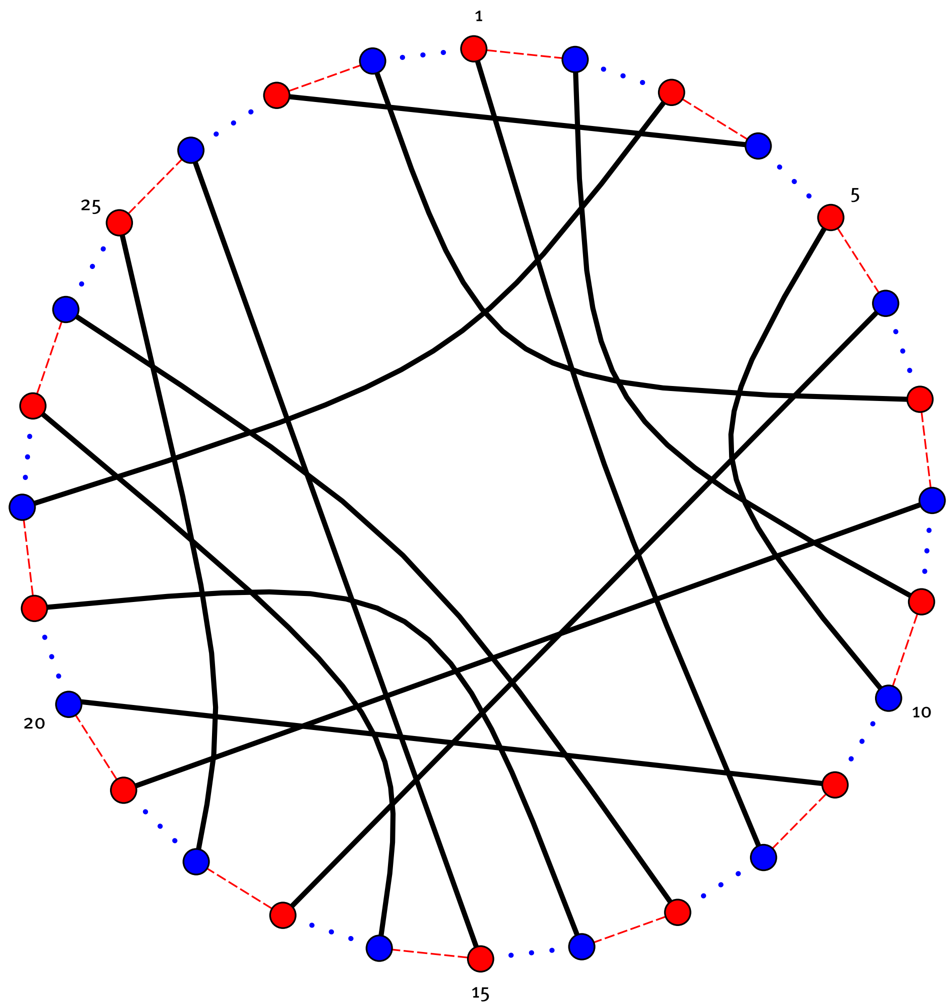


Figure 4.10.: Smallest bipartite counterexample with three pages containing perfect disjoint matchings where two of the matchings form a cycle.

and rotating we can again impose the restriction that $\mu(0)$ in the second matching is the maximal value of $\mu(\cdot)$ in both the second and the third matching.

The search space is significantly larger since we abandoned the restriction that two matchings form a cycle. Thus, we were only able to check for counterexamples up to order 18, which already took a week on the available computing hardware (4×12-Core AMD Opteron 6172, 2.1 GHz, 256 GB RAM).

We did not find any counterexample with $\leqslant 18$ vertices. That is, we can only conclude that 3-PERFECT-MATCHINGS-BOOK-EMBEDDING has a smallest bipartite counterexample with at least 20 vertices and at most 28 vertices.

Outlook

It is, therefore, a sensible extension of this work to implement a more efficient searcher or just use more computing power to get the smallest counterexample of 3-PERFECT-MATCHINGS-BOOK-EMBEDDING.

Also, the special case 3-PERFECT-MATCHINGS-BOOK-EMBEDDING may already be NP-complete. It may be possible to disprove this by getting a simple decision criterion from the structure of the counterexample in Figure 4.10. Inversely, the example may also provide a clue on how to prove the NP-hardness. This direction seems to be quite difficult since we do not really understand why this example is a counterexample.

4.3. PQ-tree on the Vertices

Besides demanding that pages have a special structure, as we have done in the preceding sections, we may restrict the order of the vertices to a subset of the symmetric group S_n that we can, hopefully, work with more easily.

Angelini et. al. [**angelini11**] showed that CONNECTED-SEFE (see page 5) can be reduced to a 2-page embedding problem where the vertex order comes from a P-tree.

For this reason it is useful to restrict the permutations with PQ-trees. That is, we do the very opposite of Section 4.1 and start with a PQ-tree instead of getting a tree that represents the possible book embeddings.

A general PQ-tree does not really help since the PQ-tree in Figure 4.11 (a single P-node) represents all permutations of $\{1, \ldots, n\}$, i. e. the problem does not get easier.

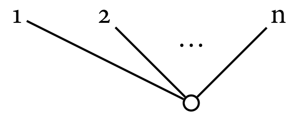


Figure 4.11.: A PQ-tree that represents all permutations of $\{1, \dots, n\}$.

Thus, we have to narrow down the possible permutations even more. In this section we only consider Q-trees and show that Q-TREE-BOOK-EMBEDDING, which is BOOK-EMBEDDING restricted to Q-trees, can be solved in quadratic time. In order to do this, we provide a reduction of the problem to 2-SAT, the problem of checking a 2-CNF formula for satisfiability. The 2-SAT problem is solvable in linear time as first shown by Krom [**Krom67**].

Problem: Q-TREE-BOOK-EMBEDDING
Given: A BOOK-EMBEDDING instance I with vertices V and a Q-tree T with leaves V.
Question: Is there a total order $< \in \pi(T)$ solving I?

Problem: P-TREE-BOOK-EMBEDDING
Given: A BOOK-EMBEDDING instance I with vertices V and a P-tree T with leaves V.
Question: Is there a total order $< \in \pi(T)$ solving I?

Problem: 2-SAT
Given: A 2-CNF Boolean formula f.
Question: Is f satisfiable?

Q-Trees are exactly the wrong type of trees compared to the reformulation of the CONNECTED-SEFE problem by Angelini et. al. [**angelini11**] since Q-nodes vastly restrict the possible permutations and are significantly easier to handle than P nodes. This section, therefore, only solves CONNECTED-SEFE if the P-tree of the equivalent P-TREE-BOOK-EMBEDDING instance is also a Q-tree, i. e. if the P-tree is a binary tree.

We first investigate what possible configurations of the leaves the book constraints lead to when we take the Q-tree T into account. Then we show how these configurations can be expressed with a 2-CNF formula.

Possible configurations resulting from a book constraint

We first show that the book embedding restrictions for two edges $\{a, b\}$ and $\{c, d\}$ can be translated directly to restrictions on the Q-tree. Before we start with this translation, however, we list some conventions. The Q-tree is called T and has leaves V. Furthermore, let $t(M)$ be the smallest subtree of T containing M and

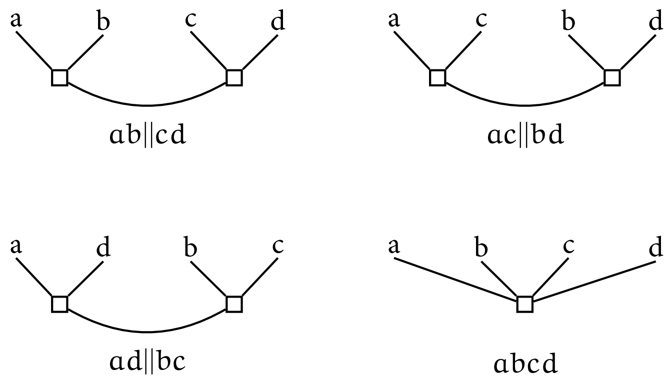


Figure 4.12.: The possible topologies of four leaves a, b, c, d in a tree.

let $r(M)$ be its root for any $M \subseteq V$. Also remember that we assumed in Section 2.1 that any two edges we consider the book constraint for are independent.

We want to distinguish cases based on which two leaves in $M := \{a, b, c, d\}$ can be separated from the others. These possible *topologies of* M *in* T are depicted in Figure 4.12. For example, we have $ab||cd$ if there is an edge $e \in E(T)$ such that a and b are in one component of $T \setminus e$ while c and d are in the other component, i. e. a and b can be separated from c and d. The topologies for $ac||bd$ and $ad||bc$ are defined analogously. If no two vertices in M can be separated from the other two (all pairs of vertices in M have the same lowest common ancestor) we say that the topology $abcd$ occurs.

Depending on which of the topologies occurs, we can map the constraint from Lemma 2.2 to a Boolean formula on the order $<$ of the vertices V.

Case 1: ab||cd

Since $\{a, b\}$ and $\{c, d\}$ are in disjoint subtrees and the vertices of a subtree are consecutive in every permutation $\pi(T)$, all tree orders fulfil the book constraint. Thus, the constraint is mapped to the Boolean expression $\mathtt{true}$.

Case 2: ac||bd

In this case we search through the possible trees to determine the resulting 2-SAT formula. To do this systematically, we have to take into account that T is ordered and rooted and determine what the tree can look like.

Thus, we further split this case into sub-cases based on whether the vertices a and c are between b and d (inside), b and d are between a and c (inside) or no two vertices

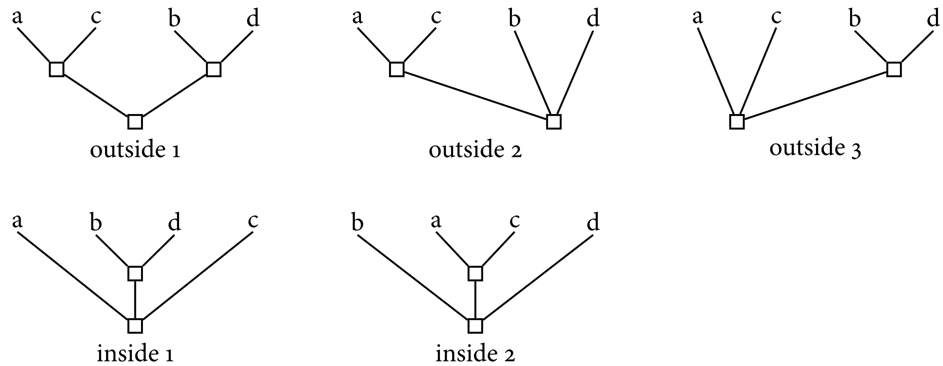


Figure 4.13.: The possible trees corresponding to $ac||bd$.

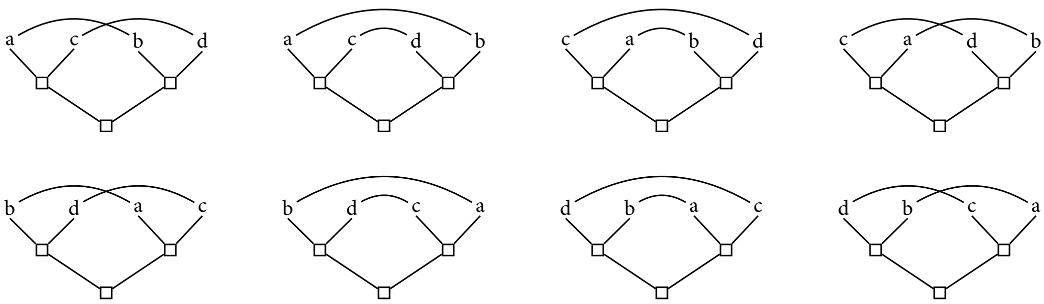


Figure 4.14.: The tree orders for the case outside 1.

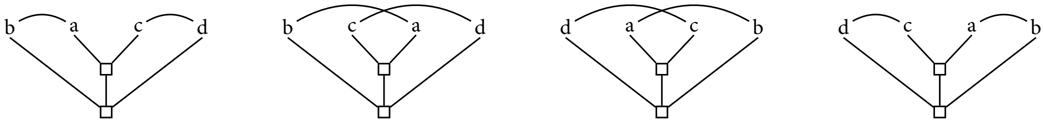


Figure 4.15.: The tree orders for the case inside 1.

in M are between the vertices they are separated from (outside). Note that a and c are between b and d for all orders in $\pi(T)$ if they are between b and d for one order in $\pi(T)$ since T is a Q-tree.

In the outside case, the possible permutations also depend on how the roots of the subtrees $t(a, c)$ and $t(b, d)$ are related, i. e. whether one appears as a child of the other. The possible tree structures are depicted in Figure 4.13.

For each tree, we can exhaustively search through the orders of the leaves M the tree permits. For the case outside 1 these orders are portrayed in Figure 4.14. We observe that the valid orders are exactly the orders with $a < c \Leftrightarrow d < b$. The other two outside cases can be handled similarly. Both of them again yield $a < c \Leftrightarrow d < b$.

For the inside cases we do the same. The possible orders of the case inside 1 are depicted in Figure 4.15. We can infer the inverse $a < c \Leftrightarrow b < d$ for both inside cases.

Case 3: ad||bc

As in case 2, we either get $a < d \Leftrightarrow b < c$ or $a < d \Leftrightarrow c < b$.

Case 4: abcd

Let r be the common root $r(a, b, c, d)$ of $M := \{a, b, c, d\}$. The tree T represents two permutations of M since the children of the Q-node r can only be reversed. If the book constraint is valid in a permutation of M, it is also valid in the mirror image of the permutation. Therefore, the book constraint may be valid in none or both of the two possible permutations. That is, we get either $\mathtt{true}$ or $\mathtt{false}$ as constraint.

Mapping BOOK-EMBEDDING to 2-SAT

We now show how the resulting Boolean expressions can be mapped to 2-SAT formulae. To do so we fix a reference orientation of the inner nodes of T. For each $\pi \in \pi(T)$ and every inner node ν in T, we can say whether we got π as a permutation in $\pi(T)$ by giving ν the reference orientation or not. Introduce a Boolean variable o_ν that stands for ν being in reference orientation.

By the construction above, a book constraint for two edges yields one of the following Boolean expressions dependent on the structure of T.

1. A trivial expression $\mathtt{true}$ or $\mathtt{false}$. (from cases 1 and 4)

2. A fixed order of two leaves v and w which is the same as fixing the order of their root $r := r(v, w)$. Thus, we get o_r or $\neg o_r$. (from case 4)
3. A connection between the orders of the leaves a, b and c, d that are located in disjoint subtrees. This is the same as tying the orders of the roots $r := r(a, b)$ and $s := r(c, d)$ together. We get either $o_r \Leftrightarrow o_s \equiv (\neg o_r \vee o_s) \wedge (\neg o_s \vee o_r)$ or $o_r \Leftrightarrow \neg o_s \equiv (\neg o_r \vee \neg o_s) \wedge (o_s \vee o_r)$. (from cases 2 and 3)

Thus, we can reduce Q-TREE-BOOK-EMBEDDING to determining whether a set of 2-CNF expressions is consistent with a Q-tree structure. But since the inner nodes of a Q-tree can be flipped completely independently of each other, the consistency with the Q-tree structure does not impose any extra restrictions. That is, Q-TREE-BOOK-EMBEDDING can be mapped to checking a 2-SAT formula for consistency (satisfiability).

We now see how the reduction from Q-TREE-BOOK-EMBEDDING to 2-SAT above can be implemented in quadratic time.

Lemma 4.7. *Q-TREE-BOOK-EMBEDDING can be reduced to 2-SAT in quadratic time.*

Proof. Let $\big((V, E_1), \dots, (V, E_k), T\big)$ be an instance of Q-TREE-BOOK-EMBEDDING. We can map the book constraints for each pair $e_1, e_2 \in E_i$ of edges for all $i \in \{1, \dots, k\}$ to a 2-CNF formula with the construction above.

Let's investigate how this can be done efficiently. Our goal is to map each book constraint resulting from a pair of edges to a 2-CNF formula in constant time after a linear time precomputation.

We assume $V = \{1, \dots, n\}$ and that each inner node of the tree T contains a pointer to its parent and an (ordered) list of its children. Furthermore, let r be the root of T.

To determine the topology of a quadruple of leaves, we need to know the lowest common ancestor of certain pairs of nodes and their initial order.

The first problem has been studied extensively. Harel and Tarjan [**Harel84**] showed the surprising result that lowest common ancestor queries can be answered in constant time after a linear time precomputation, although their algorithm was too complicated to be implemented effectively. Farach and Colton [**Farachoo**] presented a far simpler variant of this algorithm that is used in practice. We assume in the following that the precomputation has been done and that $\mathsf{LCA}(x, y)$ gives the lowest common ancestor of x and y in $\mathcal{O}(1)$ time.

For the second problem, we can precompute the index array idx of V that maps each leaf V to its index in the reference orientation of T. This can be accomplished in linear time by a simple depth-first search.

Before we begin with the actual translation, we need another helper function **Leaf-Order(a,b)** that translates a statement of the form $a < b$ for leaves $a, b \in V$ into a literal on the variable o_r where $r = \mathsf{LCA}(a, b)$. If **idx[a]** $<$ **idx[b]**, then r has reference orientation and the result is o_r. Otherwise, the result is $\neg o_r$. This decision can obviously can be made in $\mathcal{O}(1)$ time.

We now have everything we need to translate book constraints into Boolean formulae. The direct formalisation of the construction above is given in Algorithm 1. It returns a Boolean formula for a pair of edges $\{a, b\}$ and $\{c, d\}$ in $\mathcal{O}(1)$. Note that we can test whether the order **idx** fulfils the book constraint in line 9 in $\mathcal{O}(1)$ time since only the relative order of a, b, c and d is relevant.

In the algorithm, the Boolean formulae are not given in conjunctive normal form for the sake of clarity. If we want CNF formulae, we can statically replace the Boolean formulae in Algorithm 1 by their CNF equivalents.

All in all, we need $\mathcal{O}(|T|)$ time for the precomputation and $\mathcal{O}(1)$ time for each of the $\mathcal{O}(|E_1|^2 + \cdots + |E_k|^2)$ book constraints. Thus, the reduction to Q-TREE-2-SAT takes $\mathcal{O}(|T| + |E_1|^2 + \cdots + |E_k|^2)$ time. □

Since 2-SAT is solvable in linear time, we conclude that Q-TREE-BOOK-EMBEDDING can be solved in quadratic time.

Theorem 4.8. *Q-TREE-BOOK-EMBEDDING can be solved in quadratic time.*

Proof. Let $((V, E_1), \ldots, (V, E_k), T)$ be an instance of Q-TREE-BOOK-EMBEDDING. We saw that the reduction to 2-SAT takes $\mathcal{O}(|T| + |E_1|^2 + \cdots + |E_k|^2)$ time in Lemma 4.7. For each pair of edges $e_1, e_2 \in E_i$ where $i \in \{1, \ldots, n\}$ we get a 2-CNF expression of length $\mathcal{O}(1)$. Since 2-SAT is solvable in linear time as first shown by Krom [**Krom67**], we, therefore, need $\mathcal{O}(|E_1|^2 + \cdots + |E_n|^2)$ time to solve the resulting 2-SAT problem. Altogether, we need $\mathcal{O}(|T| + |E_1|^2 + \cdots + |E_k|^2)$ time.

We have assumed $|E_i| \leqslant 2|V| - 3$ for all $i \in \{1, \ldots, k\}$ at the start of Chapter 2 since the pages have to be outerplanar to be embeddable. Furthermore, the Q-tree has fewer inner nodes than its number of leaves $|V|$ since each inner node has at least two children. That is, we can rewrite the time as $\mathcal{O}(|T| + |E_1|^2 + \cdots + |E_k|^2) = \mathcal{O}(|V| + k(2|V| - 3)^2) = \mathcal{O}(k|V|^2)$.

Outlook

So book embedding is solvable in quadratic time if we constrain the vertex order by a Q-tree. But what if we have a P-tree as in the reformulation of CONNECTED-SEFE? We have already seen in Figure 4.11 that this restriction cannot make the problem

```
Input: Two edges {a, b} and {c, d}
Output: A Boolean formula representing the book constraint for the two edges
// Independent edges?
if |{a, b, c, d}| = 4 then
    r1 ← LCA(a, b)
    r2 ← LCA(a, c)
    r3 ← LCA(a, d)
    r4 ← LCA(b, c)
    r5 ← LCA(b, d)
    r6 ← LCA(c, d)
    if all ri are the same for i ∈ {1, . . . , 6} then
        // abcd
        if Order idx fulfils book constraint then
            return true
        else
            return false
    else if LCA(r2, r5) is not equal to r2 or r5 then
        // ac||bd
        if idx[a] between idx[b] and idx[d] then
            return Leaf-Order(a, c) ⇔ Leaf-Order(b, d)
        else
            return Leaf-Order(a, c) ⇔ Leaf-Order(d, b)
    else if LCA(r3, r4) is not equal to r3 or r4 then
        // ad||bc
        if idx[a] between idx[b] and idx[c] then
            return Leaf-Order(a, d) ⇔ Leaf-Order(b, c)
        else
            return Leaf-Order(a, d) ⇔ Leaf-Order(c, b)
    else
        // ab||cd
        return true
else
    return true
```

Algorithm 1: Translating the book constraint in $\mathcal{O}(1)$

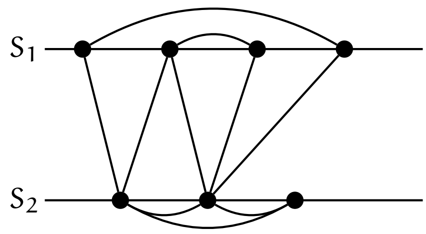


Figure 4.16.: A two-spine drawing.

simpler than the general book embedding problem. Furthermore, we cannot directly generalise our construction since the answer to "What permutation of the children of a P-node occurs?" cannot simply be modelled by a Boolean variable. That is, P-TREE-BOOK-EMBEDDING remains an interesting open problem.

4.4. Multiple Spines

In the previous sections we showed how PQ-trees relate to book embeddings. On the one hand they can help to solve the problem for connected graphs on the pages and on the other hand we can restrict the orders of the vertices with a PQ-tree, yielding an interesting modification to book embedding.

We now consider another variation on book embedding which will turn out to be a special case of the latter application of PQ-trees. It is a generalisation of the 2-page case that uses not just one spine but several parallel spines (lines) S_1, ..., S_k. In the following considerations we always assume that S_i is above S_{i+1} for all $i \in \{1, \ldots, k-1\}$. We want to planarly draw one graph above S_1, one between S_i and S_{i+1} for each $i \in \{1, \ldots, k-1\}$ and one below S_k, as depicted in Figure 4.16 for two spines. This problem is motivated by *level planarity* which is the same problem without the *caps*, the graph above S_1 and the graph below S_k. The level planarity problem was first introduced by Tomii et. al. [**Tomii77**]. Jünger, Leipert and Mutzel presented an algorithm that checks for level planarity in linear time [**Junger99**].

In this section we show that the multiple spine problem is equivalent to a 2-page book embedding problem constrained by a special P-tree, but do not manage to give an efficient algorithm it. Still, this reinforces our belief that P-TREE-BOOK-EMBEDDING is an interesting problem.

Let the spines always be $S_i = \mathbb{R} \times \{-i\}$. We now formally define the problem. It will turn out to be convenient to formally use directed edges pointing downward

Figure 4.17.: Level planarity only depends on the order of the vertices.

for the edges between the spines, but we still understand and draw these edges as undirected edges.

Problem: MULTIPLE-SPINE-EMBEDDING
Given: Vertex sets $V_1, \dots, V_k$ and edge sets $E_0 \subseteq \binom{V_1}{2}$, $E_1 \subseteq V_1 \times V_2, \dots, E_{k-1} \subseteq V_{k-1} \times V_k$, $E_k \subseteq \binom{V_k}{2}$.
Question: Is there a planar drawing of $(V_1 \cup \dots \cup V_k, E_0 \cup \dots \cup E_k)$ such that a vertex in V_i lies on S_i for all $i \in \{1, \dots, k\}$, edges do not cross a spine, the edges in E_0 lie completely above S_1 and the edges in E_k lie completely below S_k?

Tomii et. al. [**Tomii77**] showed that the 2-level planar graphs are exactly the forests of caterpillars. Recall that a *caterpillar* is a tree all of whose vertices are on a central path or one edge away from it. Therefore, each of the graphs $(V_i \cup V_{i+1}, E_i)$ for $i \in \{1, \dots, k-1\}$ has to be a forest, i. e. we find $|E_i| = |V_i| + |V_{i+1}| - l$ if this forest has l components. That is, as in the case of page embedding the number of edges is again linear in the number of vertices. Thus, the size of a MULTIPLE-SPINE-EMBEDDING instance is in $\mathcal{O}(|V_1| + \dots + |V_k|)$.

From Lemma 2.2 we know that book embedding is essentially an ordering problem. Similarly, consider two edges (a_1, b_1) and (a_2, b_2) lying between the same two spines and investigate how their embeddability depends on the order of their endpoints. If a_1 lies left of a_2 on the upper spine and b_2 lies left of b_1 on the lower spine, then any Jordan curve from a_1 to b_1 between the spines must intersect with any Jordan curve from a_2 to b_2 between the spines by the Jordan curve theorem, i. e. there cannot be a level embedding with this order. This case is depicted in Figure 4.17. Similarly, if a_2 lies left of a_1 and b_1 lies left of b_2, the edges (a_1, b_1) and (a_2, b_2) also cannot be embedded.

In any other order we can just draw a straight line for both edges to obtain a valid embedding of the edges. After combining these observations for all pairs of edges and taking the caps into account, we get a total order formulation of MULTIPLE-SPINE-EMBEDDING.

Lemma 4.9. *Let* $I := (V_1, \dots, V_k, E_0, \dots, E_k)$ *be a* MULTIPLE-SPINE-EMBEDDING *instance. Then* I *is solvable if any only if there is a linear order* $<_i$ *on* V_i *for each*

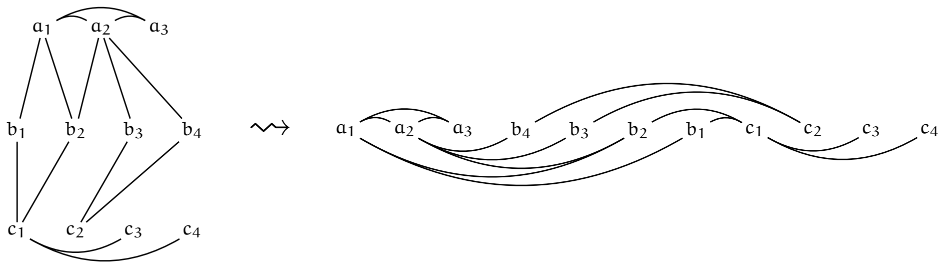


Figure 4.18.: A MULTIPLE-SPINE-EMBEDDING instance can be transformed into a 2-page book embedding instance with separated sets of vertices.

$i \in \{1, \ldots, k\}$ *such that the following properties hold. For all* $i \in \{1, \ldots, k-1\}$ *and pairs of edges* $(a_1, b_1), (a_2, b_2) \in E_i$ *the order* $a_1 <_i a_2 \wedge b_2 <_{i+1} b_1$ *does not occur. Furthermore, for* $i \in \{0, k\}$ *and all* $\{a, b\}, \{c, d\} \in E_i$ *we must not have* $a <_i c <_i b <_i d$.

The order constraint for level planarity looks very similar to the book constraint, just separated into two total orders. Indeed, if we have a MULTIPLE-SPINE-EMBEDDING instance we can find a corresponding book embedding instance.

Theorem 4.10. *Let* $I := (V_1, \ldots, V_k, E_0, \ldots, E_k)$ *be a MULTIPLE-SPINE-EMBEDDING instance. We define a corresponding 2-page book embedding instance by taking* $V := V_1 \cup \cdots \cup V_k$ *as vertices and*

$$\widetilde{E}_1 := \bigcup_{\substack{i \in \{0, \ldots, k\} \\ i \text{ even}}} E_i$$

$$\widetilde{E}_2 := \bigcup_{\substack{i \in \{0, \ldots, k\} \\ i \text{ odd}}} E_i$$

as pages. Then I *is solvable if and only if* $J := (V, \widetilde{E}_1, \widetilde{E}_2)$ *has a book embedding where the vertices in each* V_i *are consecutive.*

Proof.

"⇒" Let $<_i$ for $i \in \{1, \ldots, k\}$ be total orders forming a valid embedding of I.

Then define $<$ on V to be the total order that first lists the vertices of V_1, then the vertices of V_2, and so on. Get the inner order of the vertices in V_i from $<_i$ if i is even and from $<_i$ reversed if i is odd. This construction is illustrated in Figure 4.18.

The order $<$ is a valid solution of the book embedding problem J: The edges in distinct edge sets E_i do not intersect by the construction of $<$ and the definitions of $\widetilde{E}_1$ and $\widetilde{E}_2$. Edges in E_0 or E_k do not intersect since $<_1$ and $<_k$ are valid page embeddings for (V_1, E_0) and (V_k, E_k), respectively. Now take two edges $(a_1, b_1), (a_2, b_2) \in E_i$ for some $i \in \{1, \ldots, k-1\}$. If $a_1 < a_2 < b_1 < b_2$ occurs, we have $a_1 <_i a_2 \ \wedge \ b_1 <_{i+1} b_2$, contradicting the validity of the initial solution of I. Thus, the book constraint for the two edges is fulfilled.

"$\Leftarrow$" Let $<$ be a valid book order of J where the sets V_i with $i \in \{1, \ldots, k\}$ are separated. Do the construction above in reverse, i. e. define $<_i$ to be the restriction of $<$ to V_i for all $i \in \{1, \ldots, k\}$. Additionally, reverse $<_i$ when i is odd.

The order $<_i$ yields a valid embedding for I: The caps already appeared in the book embedding problem J, i. e. they are still valid. If $a_1 <_i a_2 \ \wedge \ b_2 <_{i+1} b_1$ occurs for some $(a_1, b_1), (a_2, b_2) \in E_i$ and $i \in \{1, \ldots, k-1\}$, then we must have either $a_1 < a_2 < b_1 < b_2$, $a_2 < a_1 < b_2 < b_1$, $b_1 < b_2 < a_2 < a_1$ or $b_2 < b_1 < a_1 < a_2$. All of these cases contradict the book constraints. $\square$

Outlook

All in all, we see that MULTIPLE-SPINE-EMBEDDING is equivalent to a 2-page book embedding problem where the vertex sets V_i with $i \in \{1, \ldots, k\}$ have to be separated. This separation can be modelled by a P-tree by introducing a P-node connected to the vertices V_i for all $i \in \{1, \ldots, k\}$ and connecting all of these P-nodes to a single root.

Since MULTIPLE-SPINE-EMBEDDING is an interesting problem in its own right, this leaves several distinct possibilities for further results:

- Provide a polynomial time algorithm for P-TREE-BOOK-EMBEDDING and get an efficient solution of MULTIPLE-SPINE-EMBEDDING.
- Prove the NP-completeness of MULTIPLE-SPINE-EMBEDDING and get the NP-completeness of P-TREE-BOOK-EMBEDDING.
- Provide a polynomial time algorithm for MULTIPLE-SPINE-EMBEDDING and get an efficient algorithm for a special case of P-TREE-BOOK-EMBEDDING.

5. Conclusion

In this thesis we considered the book embedding problem where the assignment of edges to pages has already been fixed.

We proved that BOOK-EMBEDDING is NP-complete for a linear number of pages in Chapter 3, even if the pages are matchings. In the same chapter we showed how BOOK-EMBEDDING can still be solved in super-polynomial time by expressing it with 3-CNF-formulae. Though matchings are a nicely restricted case that is already NP-complete, it is dissatisfying that we need an unbounded number of pages for our NP-hardness proof. We would like to show NP-completeness for a constant number of pages similar to the general book embedding problem, which is NP-complete for two pages [**Bernhart79**]. The problem BOOK-EMBEDDING may be NP-complete for the next smaller case of three pages, but proving or disproving that seems to be quite difficult.

The remainder of the work was concerned with a variety of special cases and restrictions of BOOK-EMBEDDING in Chapter 4. We first considered pages containing connected graphs and showed that embeddability can be decided in linear time in this case by representing all possible book embeddings using a PQ-tree.

Next, we dealt with the very opposite with regards to connectivity: the pages are disjoint perfect matchings. We showed that bipartiteness is necessary for embeddability in this case and provided bipartite examples and counterexamples for all numbers of pages except for three pages. We computed that the smallest counterexample for three pages has at least 20 vertices and at most 28. When two matchings form a cycle, we found a smallest counterexample of order 28. This is too large for us to be able to infer anything useful from it. One obvious extension of this case is to find some structure in the counterexamples even though they are large and, maybe, get a better necessary condition or a good sufficient condition. The counterexample for three pages may also yield a clue on whether BOOK-EMBEDDING is NP-complete for three pages.

The problem that we considered after that was to restrict the order of the vertices on the spine by a Q-tree. We showed that the book constraints turn into simple constraints on the Q-tree in this case. This allowed us to solve the problem in quadratic time. The most interesting continuation of this line of thought is to make the restriction more in accordance with its motivation. That is, to use P-trees as in the reduction of CONNECTED-SEFE to a 2-page P-TREE-BOOK-EMBEDDING instance by Angelini et. al. [**angelini11**]. We already argued that this does not make the problem easier than BOOK-EMBEDDING since a P-tree can represent all permutations on its leaves. Still, maybe we can get a solution for just two pages which is all that is needed for solving CONNECTED-SEFE.

Finally, we varied the book embedding problem by allowing multiple spines. We showed that this case is equivalent to a restricted 2-page P-TREE-BOOK-EMBEDDING instance. Although this did not efficiently solve the problem, it provided us with several future extensions:

- Provide a polynomial time algorithm for P-TREE-BOOK-EMBEDDING and get an efficient solution of MULTIPLE-SPINE-EMBEDDING.
- Prove the NP-completeness of MULTIPLE-SPINE-EMBEDDING and get the NP-completeness of P-TREE-BOOK-EMBEDDING.
- Provide a polynomial time algorithm for MULTIPLE-SPINE-EMBEDDING and get an efficient algorithm for a special case of P-TREE-BOOK-EMBEDDING.

The last extension may also give some helpful pointers on how to approach the general P-TREE-BOOK-EMBEDDING problem.

All in all, the most important continuation of this work is to find the computational complexity of two problems: BOOK-EMBEDDING for a constant number of pages and P-TREE-BOOK-EMBEDDING. In the following we list possible approaches and sub-problems that could possibly be of use, ordered decreasingly by how likely we believe the approach to succeed or how useful the sub-problem is:

1. Prove the NP-completeness of BOOK-EMBEDDING for a constant number of pages:
    a) Compute a smallest bipartite counterexample of 3-PERFECT-MATCHINGS-BOOK-EMBEDDING.

b) Show that 3-PERFECT-MATCHINGS-BOOK-EMBEDDING is NP-complete by looking at the structure of Figure 4.10 or derive a necessary and sufficient condition from it that is efficiently checkable.

2. Find the computational complexity of P-TREE-BOOK-EMBEDDING:

   a) Give an efficient algorithm for MULTIPLE-SPINE-EMBEDDING or show that it is NP-complete.

   b) Solve CONNECTED-SEFE.

   c) Generalise the approach of Section 4.3 to P-trees.

Appendix

A. Symbols and Notations

$\mathbb{N}$	set of natural numbers $\mathbb{N} := \{1, 2, \dots\}$
$\mathbb{Z}$	set of integers $\mathbb{Z} := \{\dots, -1, 0, 1, \dots\}$
$\mathbb{R}$	set of real numbers
$\mathbb{R}_{\geqslant 0}$	set of nonnegative real numbers
$\lvert x\rvert$	absolute value of real number x
$\lvert M\rvert$	cardinality of set M
$\mathrm{Sym}(M)$	set of permutations on M
$\lvert G\rvert$	order of the graph G, i. e. its number of vertices
$f = \mathcal{O}(g)$	function $f\colon \mathbb{N} \to \mathbb{R}_{\geqslant 0}$ grows asymptotically at most as fast as function $g\colon \mathbb{N} \to \mathbb{R}_{\geqslant 0}$, i. e. there is a $n_0 \in \mathbb{N}$ and a $c \in \mathbb{R}_{\geqslant 0}$ such that $f(n) \leqslant cg(n)$ for all $n \geqslant n_0$
$\mathcal{C}(X, Y)$	set of continuous functions from X to Y
$A \cap B$	intersection of sets or graphs A and B
$A \cup B$	union of sets or graphs A and B
$M \times N$	set of pairs (m, n) with $m \in M$ and $n \in N$
$\binom{V}{k}$	set of k-element subsets of V, $\binom{V}{k} := \{U \subseteq V : \lvert U\rvert = k\}$
C_n	cycle on n vertices
K_n	complete graph on n vertices
$K_{m,n}$	complete bipartite graph with m left vertices and n right vertices
$\wedge$	logical and
$\vee$	logical or
$\dot\vee$	logical exclusive or
CNF	conjunctive normal form
$x \equiv y$	Boolean formula x is equivalent to Boolean formula y
$P_1 \leqslant_P P_2$	problem P_1 admits a polynomial time reduction to problem P_2
TM	Turing machine
P	problems solvable by a deterministic TM in polynomial time
NP	problems solvable by a non-deterministic TM in polynomial time

B. List of Figures

C. List of Problems

D. List of Tables

E. List of Algorithms