A Tight Subexponential-Time Algorithm for Two-Page Book Embedding

Robert Ganian
Algorithms and Complexity Group, TU Wien, Austria

Haiko Müller
School of Computing, University of Leeds, UK

Sebastian Ordyniak
School of Computing, University of Leeds, UK

Giacomo Paesani
School of Computing, University of Leeds, UK

Mateusz Rychlicki
School of Computing, University of Leeds, UK

Abstract

A book embedding of a graph is a drawing that maps vertices onto a line and edges to simple pairwise non-crossing curves drawn into "pages", which are half-planes bounded by that line. Two-page book embeddings, i.e., book embeddings into 2 pages, are of special importance as they are both NP-hard to compute and have specific applications. We obtain a $2^{\mathcal{O}(\sqrt{n})}$ algorithm for computing a book embedding of an n-vertex graph on two pages – a result which is asymptotically tight under the Exponential Time Hypothesis. As a key tool in our approach, we obtain a single-exponential fixed-parameter algorithm for the same problem when parameterized by the treewidth of the input graph. We conclude by establishing the fixed-parameter tractability of computing minimum-page book embeddings when parameterized by the feedback edge number, settling an open question arising from previous work on the problem.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases book embedding, page number, subexponential algorithms, subhamiltonicity, feedback edge number

Digital Object Identifier 10.4230/LIPIcs.ICALP.2024.68

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/abs/2404.14087` [24]

Funding *Robert Ganian*: Project No. Y1329 of the Austrian Science Fund (FWF), Project No. ICT22-029 of the Vienna Science Foundation (WWTF).
Sebastian Ordyniak: Project EP/V00252X/1, EPSRC.

1 Introduction

Book embeddings of graphs are drawings centered around a line, called the *spine*, and half-planes bounded by the spine, called *pages*. In particular, a k-page book embedding of a graph G is a drawing which maps vertices to distinct points on the spine and edges to simple curves on one of the k pages such that no two edges on the same page cross [6]. These embeddings have been the focus of extensive study to date [16, 20, 21, 22, 25, 38, 47], among others due to their classical applications in VLSI, bio-informatics, and parallel computing [11, 20, 31].

Every n-vertex graph is known to admit an $\lceil \frac{n}{2} \rceil$-page book embedding [6, 11, 30], but in many cases it is possible to obtain book embeddings with much fewer pages. Particular attention has been paid to two-page embeddings, which have specifically been used, e.g., to represent RNA pseudoknots [31, 42]. The class of graphs that can be embedded on two pages was studied by Di Giacomo and Liotta [27], Heath [32] as well as by other authors [1], and was shown to be a superclass of planar graphs with maximum degree at most 4 [5].

While two-page book embeddings are a special class of planar embeddings, they are not polynomial-time computable unless $\mathsf{P} = \mathsf{NP}$. Indeed, a graph admits a two-page book embedding if and only if it is *subhamiltonian* (*i.e.*, is a subgraph of a planar Hamiltonian graph) [6] and testing subhamiltonicity is an NP-hard problem [11]. On the other hand, the aforementioned problem of constructing a two-page book embedding (or determining that none exists) – which we hereinafter call Two-Page Book Embedding – becomes linear-time solvable if one is provided with a specific ordering of the n vertices of the input graph along the spine [31]. While Two-Page Book Embedding can be seen to admit a trivial brute-force $2^{\mathcal{O}(n \cdot \log n)}$ algorithm, it has also been shown to be solvable in $2^{\mathcal{O}(n)}$ time – in particular, one can branch to determine the allocation of edges into the two pages and then solve the problem via dynamic programming on SPQR trees [2, 33, 34].

Contribution. As our main contribution, we break the single-exponential barrier for Two-Page Book Embedding by providing an algorithm that solves the problem in $2^{\mathcal{O}(\sqrt{n})}$ time. Our algorithm is exact and deterministic, and avoids the single-exponential overhead of branching over edge allocations to pages by instead attacking the equivalent subhamiltonicity testing formulation of the problem. It is also asymptotically optimal under the Exponential Time Hypothesis [35]: there is a well-known quadratic reduction that excludes any $2^{o(\sqrt{n})}$ algorithm for Hamiltonian Cycle on cubic planar graphs [26], and a linear reduction from that problem (under the same restrictions) to subhamiltonicity testing [46] then excludes any $2^{o(\sqrt{n})}$ algorithm for our problem of interest.

The central component of our result is a non-trivial dynamic programming procedure that solves Two-Page Book Embedding in time $2^{\mathcal{O}(tw)} \cdot n$, where tw is the treewidth of the input graph. The desired subexponential algorithm then follows by the well-known fact that n-vertex planar graphs have treewidth at most $\mathcal{O}(\sqrt{n})$ [28, 39, 44]. But in addition to that, we believe our single-exponential treewidth-based algorithm to be of independent interest also in the context of parameterized algorithmics [13, 19].

Indeed, while Two-Page Book Embedding was already shown to be fixed-parameter tractable w.r.t. treewidth (i.e., to admit an algorithm running in time $f(tw) \cdot n$) by Bannister and Eppstein [3], that result crucially relied on Courcelle's Theorem [12]. More specifically, they showed that the required property can be encoded via a constant-size sentence in Monadic Second Order logic, which suffices for fixed-parameter tractability – but unfortunately not for a single-exponential algorithm, and a direct dynamic programming algorithm based on the characterization employed there seems to necessitate a parameter dependency that is more than single-exponential. Moreover, it is not at all obvious how one could employ convolution-based tools – which have successfully led to $2^{\mathcal{O}(tw)} \cdot n$ algorithms for, e.g., Hamiltonian Cycle [10, 14, 15] – for our problem of interest here.

Instead, we obtain our results by employing dynamic programming along a *sphere-cut decomposition* – a type of branch decomposition specifically designed for planar graphs of small treewidth [18]. However, unlike in previous applications of sphere-cut decompositions [36, 40], our algorithm requires the nooses delimiting the bags in the sphere-cut decomposition to admit a fixed drawing since our arguments rely on constructing a hypothetical solution (a subhamiltonian curve) that is "well-behaved" w.r.t. a fixed set of curves. While this would typically lead to extensive case analysis to compute the records of a parent noose from the records of the children, we introduce a generic framework that allows us to transfer records from child to parent nooses via XOR operations. We believe that this technique may be of broader interest, specifically when working with problems which require one to enhance the embedding or drawing of an input graph.

In the final part of the article, we turn our attention to the parameterized complexity of computing book embeddings. While Two-Page Book Embedding is fixed-parameter tractable when parameterized by the treewidth of the input graph, the only graph parameter which has been shown to yield fixed-parameter algorithms for computing ℓ-page book embeddings for $\ell > 2$ is the *vertex cover number*[1] [7]. Whether this tractability result also holds for other structural graph parameters such as treewidth, *treedepth* [41] or the *feedback edge number* [45] has been stated as an open question in the field[2]. We conclude by providing a novel fixed-parameter algorithm for computing ℓ-page book embeddings (or determining that one does not exist) under the third parameterization mentioned above – the feedback edge number, i.e., the edge deletion distance to acyclicity. This result is complementary to the known vertex-cover based fixed-parameter algorithm, and can be seen as a necessary stepping stone towards eventually settling the complexity of computing ℓ-page book embeddings parameterized by treewidth. Moreover, since the obtained kernel is linear in the case of $\ell = 2$, the obtained kernel allows us to generalize our main algorithmic result to a run-time of $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ where k is the feedback edge number of the input graph.

2 Preliminaries

Basic Notions. We use basic terminology for graphs and multi-graphs [17], and assume familiarity with the basic notions of parameterized complexity and fixed-parameter tractability [13, 19]. The *feedback edge number* of G, denoted by $\mathsf{fen}(G)$, is the minimum size of any *feedback edge set* of G, *i.e.*, a set $F \subseteq E(G)$ such that $G - F = (V(G), E(G) \setminus F)$ is acyclic.

For a face f of a plane graph, we use $\sigma(f)$ to denote the cyclic sequence of the vertices obtained by traversing the closed curve representing the border of f in a clock-wise manner.

Book Embeddings and Subhamiltonicity. An ℓ-page book embedding of a multi-graph $G = (V, E)$ will be denoted by a pair $\langle \prec, \sigma \rangle$, where $\prec$ is a linear order of V, and $\sigma \colon E \to [\ell]$ is a function that maps each edge of E to one of ℓ pages $[\ell] = \{1, 2, \ldots, \ell\}$. In an ℓ-page book embedding $\langle \prec, \sigma \rangle$ it is required that for no pair of edges $uv, wx \in E$ with $\sigma(uv) = \sigma(wx)$ the vertices are ordered as $u \prec w \prec v \prec x$, i.e., each page must be crossing-free. The *page number* of a graph G is the minimum number ℓ such that G admits an ℓ-page book embedding. The general problem of computing the page number of an input graph is thus:

> Book Thickness
> *Instance:* A multi-graph G with n vertices and a positive integer ℓ.
> *Question:* Does G admit a ℓ-page book embedding?

It is known that a multi-graph admits a 2-page book embedding if and only if it is *subhamiltonian*, i.e., if it has a planar Hamiltonian supergraph on the same vertex set [6]; an illustration is provided in Figure 1. Hence, the problem of deciding whether a multi-graph has page number 2 can be equivalently stated as:

---

1 The vertex cover number is the minimum size of a vertex cover, and represents a much stronger restriction on the structure of the input graphs than, e.g., treewidth.

2 E.g., at **Advances in Parameterized Graph Algorithms** (Spain, May 2–7 2022) and also at Dagstuhl seminar 21293 **Parameterized Complexity in Graph Drawing** [23].

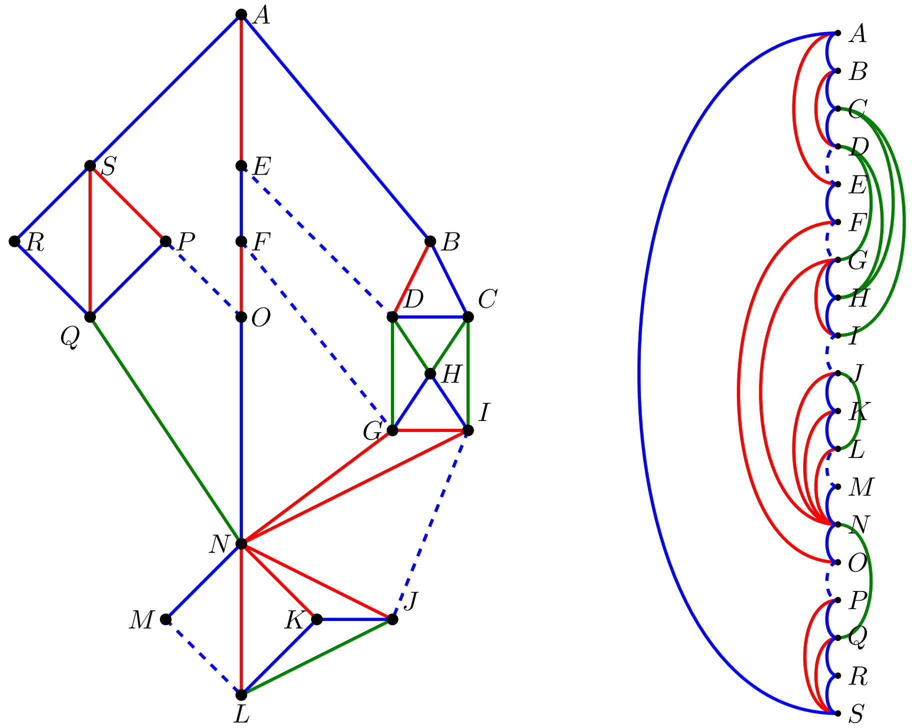

Figure 1 A drawing of a subhamiltonian graph G, made of the full-edges, which is completed by the dashed edges to one of its Hamiltonian supergraphs G_H (left) and the same graph drawn as a two-page book embedding (right). In both drawings the Hamiltonian cycle H is colored in blue and the edges belonging to page 1 and 2 are colored with green and red, respectively.

Subhamiltonicity (SUBHAM)	
Instance:	A multi-graph G with n vertices.
Question:	Is G subhamiltonian?

Since the transformation between 2-page book embeddings and Hamiltonian cycles of supergraphs is constructive in both directions, a constructive algorithm for SUBHAM (such as the one presented here) allows us to also output a 2-page book embedding for the graph.

Let G be subhamiltonian. For a Hamiltonian cycle H on $V(G)$ (where H is not necessarily a subgraph of G), we denote by G_H the graph obtained from G after adding the edges of H and say that H is a *witness* for G if G_H is planar. A drawing D of G *respects* H if D can be completed to a planar drawing of G_H by only adding the edges of H. We extend the notion of "witness" to include all the information defining the solution as follows: a tuple (D, D_H, G_H, H) is a *witness* for G if G_H is a planar supergraph of G containing the Hamiltonian cycle H, D_H is a planar drawing of G_H, and D is the restriction of D_H to G; note that D_H witnesses that D respects H.

SPQR-Trees. We assume familiarity with the SPQR-tree data structure for biconnected multi-graphs which decomposes a graph into (S)eries, (P)arallel, (R)igid and (Q) nodes (leaf nodes and root node), following the formalism used by Gutwenger et al. [29], see also [4, 8, 9]. For a node b in an SPQR-tree, we use $\textsc{Sk}(b)$ and $\textsc{Pe}(B)$ to denote the *skeleton* and *pertinent graph* of b, respectively. SPQR-trees can be computed in linear time, and an illustration of the data structure is provided in Figure 2.

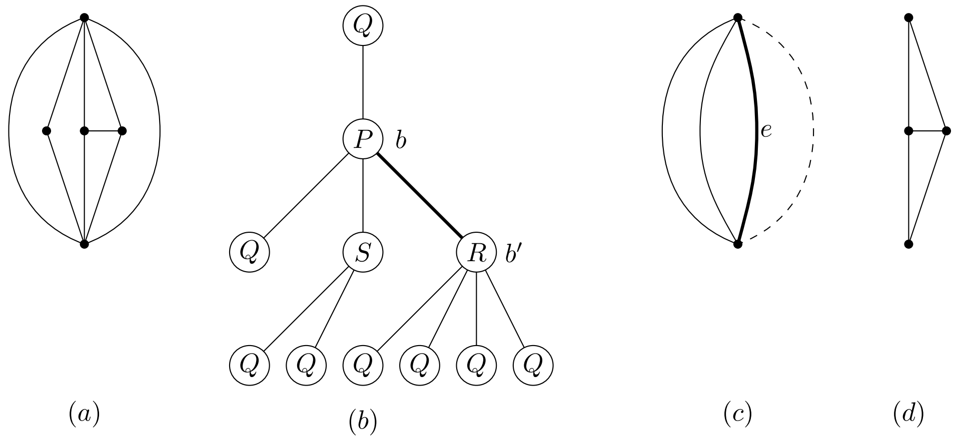

Figure 2 (a) shows a biconnected multi-graph G. (b) shows the SPQR-tree $\mathcal{B}$ of G. (c) shows the skeleton of b, $\textsc{Sk}(b)$, where the edge e that corresponds to the child (with pertinent node) b' is in bold and the dashed edge represents the reference edge. Finally, (d) shows $\textsc{Pe}(b')$.

Sphere-Cut Decompositions. A branch decomposition $\langle T, \lambda\rangle$ of a graph G consists of an unrooted ternary tree T (meaning that each node of T has degree one or three) and of a bijection $\lambda : \mathcal{L}(T) \leftrightarrow E(G)$ from the leaf set $\mathcal{L}(T)$ of T to the edge set $E(G)$ of G; to distinguish $E(T)$ from $E(G)$, we call the elements of the former *arcs* (as was also done in previous work [18]). For each arc a of T, let T_1 and T_2 be the two connected components of $T - a$, and, for $i = 1, 2$, let G_i be the subgraph of G that consists of the edges corresponding to the leaves of T_i, i.e., the edge set $\{\lambda(\mu) : \mu \in \mathcal{L}(T) \cap V(T_i)\}$. The middle set $mid(a) \subseteq V(G)$ is the intersection of the vertex sets of G_1 and G_2, i.e., $mid(a) := V(G_1) \cap V(G_2)$. The width $\beta(\langle T, \lambda\rangle)$ of $\langle T, \lambda\rangle$ is the maximum size of the middle sets over all arcs of T, i.e., $\beta(\langle T, \lambda\rangle) := max\{|mid(a)| : a \in E(T)\}$. An optimal branch decomposition of G is a branch decomposition with minimum width; this width is called the branchwidth $\beta(G)$ of G. We will need the following well-known relation between treewidth and branchwidth.

▶ **Lemma 1** ([43, Theorem 5.1]). *Let G be a graph. Then, $\mathrm{bw}(G) - 1 \leq tw(G) \leq \frac{3}{2}\mathrm{bw}(G) - 1$, where $\mathrm{bw}(G)$ is the branchwidth and $\mathrm{tw}(G)$ is the treewidth of G.*

Let D be a plane drawing of a connected planar graph G. A noose of D is a closed simple curve that (i) intersects D only at vertices and (ii) traverses each face at most once, i.e., its intersection with the region of each face forms a connected curve. The length of a noose is the number of vertices it intersects, and every noose O separates the plane into two regions δ_1 and δ_2. A *sphere-cut decomposition* $\langle T, \lambda, \Pi = \{\, \pi_a \mid a \in E(T)\,\}\rangle$ of (G, D) is a branch decomposition$\langle T, \lambda\rangle$ of G together with a set Π of circular orders π_a of $mid(a)$ – one for each arc a of T – such that there exists a noose O_a whose closed discs δ_1 and δ_2 enclose the drawing of G_1 and of G_2, respectively. Observe that O_a intersect G exactly at $mid(a)$ and its length is $|mid(a)|$. Note that the fact that G is connected together with Conditions (i) and (ii) of the definition of a noose implies that the graphs G_1 and G_2 are both connected and that the set of nooses forms a laminar set family, that is, any two nooses are either disjoint or nested. A clockwise traversal of O_a in the drawing of G defines the cyclic ordering π_a of $mid(a)$. We always assume that the vertices of every middle set $mid(a)$ are enumerated according to π_a. A sphere-cut decomposition of a given planar graph with n vertices can be constructed in $\mathcal{O}(n^3)$ time [18].

We say that a biconnected planar multi-graph G equipped with an SPQR-tree $\mathcal{B}$ is *associated* with a set $\mathcal{T}$ of sphere-cut decompositions if $\mathcal{T}$ contains a sphere-cut decomposition of $\textsc{Sk}(b)$ for every R-node and every S-node b of $\mathcal{B}$.

▶ **Lemma 2.** *Let G be biconnected planar multi-graph with planar drawing D and SPQR-tree $\mathcal{B}$ of G together with the associated set $\mathcal{T}$ of sphere-cut decompositions. Then, D can be extended to a planar drawing D' of G together with all nooses in $\{ O_a \mid a \in E(T_b) \wedge \langle T_b, \lambda_b, \Pi_b \rangle \in \mathcal{T} \}$ as well as a noose N_b for every node b of $\mathcal{B}$ satisfying:*

- *N_b intersects with D only at s_b and t_b.*
- *N_b separates $\textsc{Pe}(b)$ from $G \setminus \textsc{Pe}(b)$ in D.*

Moreover, if any of the subcurves of the nooses O_a and the nooses N_b connect the same two vertices in the same face of D, then the two subcurves are identical in D'.

Non-Crossing Matchings. Let K_n be the complete graph on vertices $\{1, \dots, n\}$ and let $<$ be a cyclic ordering of the elements in $\{1, \dots, n\}$. A *non-crossing matching* is a matching M in the graph K_n such that for every two edges $\{a, b\}, \{c, d\} \in M$ it is not the case that $a < c < b < d$.

3 Solution Normal Form

Our first order of business is to show that we can assume that the solution (Hamiltonian cycle) to the SUBHAM problem interacts with the drawing in a restricted manner. In particular, we aim to show that every subhamiltonian graph G has a witness (D, D_H, G_H, H) in *normal form*, i.e., with the following property: it is possible to draw a curve in D_H between any two vertices occurring in a common face of D such that this curve only crosses the Hamiltonian cycle at most twice. Note that this property will allow us to bound the number of possible interactions of the Hamiltonian cycle with any subgraph corresponding to either a node in the SQPR-tree or an arc in a sphere-cut decomposition and is crucial to bound the number of types in our dynamic programming algorithm. The following lemma is the main technical lemma behind our normal form. An illustration of the main ideas behind the proof is provided in Figure 3.

▶ **Lemma 3.** *Let G be a subhamiltonian graph with witness (D, D_H, G_H, H), let f be a face of D and let c be a curve drawn inside f between two vertices $u, v \in V(f)$. Then, there is a witness $(D, D_{H'}, G_{H'}, H')$ for G such that:*

(1) *$D_{H'}$ and D_H differ only inside f.*
(2) *c crosses at most two curves corresponding to the edges of H'.*
(3) *c crosses each curve corresponding to an edge of H' at most once.*

We are now ready to define our normal form for the Hamiltonian cycle. Essentially, we show that if there is a Hamiltonian cycle, then there is one which crosses each subcurve that is either part of the border of a node in the SPQR-tree or that is a subcurve of some noose in a sphere-cut decomposition of an R-node or an S-node at most twice.

Let G be a biconnected subhamiltonian multi-graph with SPQR-tree $\mathcal{B}$ and the associated set $\mathcal{T}$ of sphere-cut decompositions $\langle T_b, \lambda_b, \Pi_b \rangle$ of $\textsc{Sk}(b)$ for every R-node and S-node b of $\mathcal{B}$. We say that a witness $W = (D, D_H, G_H, H)$ for G *respects* the sphere-cut decompositions in $\mathcal{T}$, if there is a planar drawing of all nooses in the sphere-cut decompositions of $\mathcal{T}$ into D such that every subcurve c in $\bigcup_{a \in E(T_b)} O_a$ crosses the curves corresponding to the edges of H at most twice in D_H. We say that the witness W for G *respects* $\mathcal{B}$ if it respects the sphere-cut decompositions in $\mathcal{T}$ and for every node b of $\mathcal{B}$ with reference edge (s_b, t_b), it holds that there is a noose N_b that can be drawn into D_H such that:

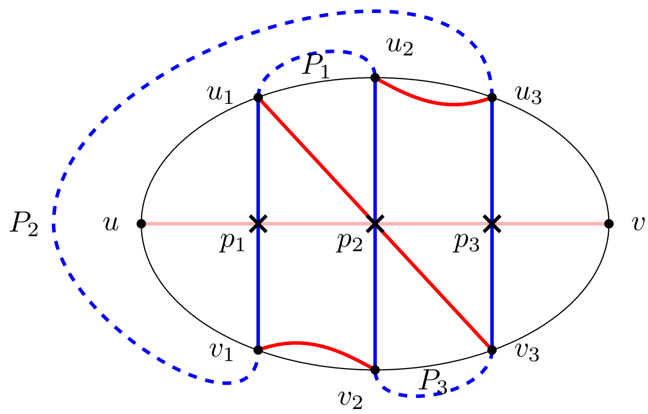

Figure 3 The cycle $H = (u_2, P_1, u_1, v_1, P_2, u_3, v_3, P_3, v_2, u_2)$ represents a Hamiltonian cycle that crosses the uv-curve at least three times (in p_1, p_2 and p_3). Thanks to Lemma 3, we obtain a Hamiltonian cycle $H' = (u_2, P_1, u_1, v_3, P_3, v_2, v_1, P_2, u_3, u_2)$ that differs from H only inside the face $f = (u, u_1, u_2, u_3, v, v_3, v_2, v_1)$ and crosses the uv-curve two fewer times than H does. Finally, note that the vertices u and v are part of either P_1, P_2, or P_3.

- N_b touches D only at s_b and t_b.
- N_b separates $\text{Pe}(b)$ from $G \setminus \text{Pe}(b)$ in D.
- Each of the two subcurves L_b and R_b obtained from N_b by splitting N_b at s_b and t_b crosses the curves corresponding to the edges of H at most twice.
- Moreover, if any of the subcurves of the nooses O_a and the nooses N_b connect the same two vertices in the same face of D, then the two subcurves are identical.

The following lemma allows us to assume our normal form and follows easily from a repeated application of Lemma 3.

▶ **Lemma 4.** *Let G be a biconnected subhamiltonian multi-graph with SPQR-tree $\mathcal{B}$ and the associated set $\mathcal{T}$ of sphere-cut decompositions. Then, there is a witness $W = (D, D_H, G_H, H)$ for G that respects $\mathcal{B}$.*

4 Setting Up the Framework

In this section we provide the foundations for our algorithm. That is, in Subsection 4.1, we show that it suffices to consider biconnected graphs allowing us to employ SPQR-trees. We then define the types for nodes in the SPQR-tree, which we compute in our dynamic programming algorithm on SPQR-trees, in Subsection 4.2. Finally, in Subsection 4.3 we introduce our general framework for simplifying dynamic programming algorithms on sphere-cut decompositions and introduce the types for nodes of a sphere-cut decomposition.

4.1 Reducing to the Biconnected Case

We begin by showing that any instance of SUBHAM can be easily reduced to solving the same problem on the biconnected components of the same instance. It is well-known that SUBHAM can be solved independently on each connected component of the input graph, the following theorem now also shows that the same holds for the biconnected components of the graph and allows us to employ SPQR-trees for our algorithm.

▶ **Theorem 5.** *Let G be a graph and let $C \subseteq V(G)$ such that $N(C) = \{n\}$, where $N(C) = \{\, v \in V(G) \setminus C \mid \exists c \in C \; \{v, c\} \in E(G) \,\}$ is the set of neighbors of any vertex of C in $V(G) \setminus C$. Then G is subhamiltonian if and only if both $G^- = G - C$ and $G^C = G[C \cup \{n\}]$ are subhamiltonian.*

4.2 Defining the Types for Nodes in the SPQR-tree

Here, we define the types for nodes in the SPQR-tree that we will later compute using dynamic programming. In the following, we assume that G is a biconnected multi-graph with SPQR-tree $\mathcal{B}$ and the associated set $\mathcal{T}$ of sphere-cut decompositions. Let b be a node of $\mathcal{B}$ with pertinent graph $\textsc{Pe}(b)$ and reference edge $e = (s, t)$. A *type* of b is a triple (ψ, M, S) such that (please refer also to Figure 4 for an illustration of some types):

- ψ is a function from $\{L, R\}$ to subsets of $\{l, l', r, r'\}$ such that $\psi(L) \in \{\emptyset, \{l\}, \{l, l'\}\}$ and $\psi(R) \in \{\emptyset, \{r\}, \{r, r'\}\}$. We denote by $V(\psi)$ the set $\psi(L) \cup \psi(R)$. Informally, ψ captures how many times the Hamiltonian cycle enters and exits the graph $\textsc{Pe}(b)$ from the left (L) and from the right (R).
- $M \subseteq \{\, \{u, v\} \mid u, v \in \{s, t\} \cup V(\psi) \wedge u \neq v \,\}$ and M is a non-crossing matching w.r.t. the circular ordering (s, r, r', t, l', l) that matches all vertices in $V(\psi)$ (i.e. $V(\psi) \subseteq V(M)$), where $V(M) = \bigcup_{e \in M} e$. Informally, M captures the maximal path segments of the Hamiltonian cycle inside $\textsc{Pe}(b) \cup V(\psi)$ with endpoints in $\{s, t\} \cup V(\psi)$.
- $S \subseteq \{s, t\} \setminus V(M)$. Informally, S captures whether s or t are contained as inner vertices on path segments corresponding to M.

We now provide the formal semantics of types; see Figure 4 for an illustration. Let $\mathcal{X}$ be the set of all types and $\textsc{Pe}^*(b)$ be the graph obtained from $\textsc{Pe}(b)$ after adding the dummy vertices l, l', r, and r' together with the edges sl, ll', $l't$, sr, rr', and $r't$. We say that b has type $X = (\psi, M, S)$ if there is a set $\mathcal{P}$ of vertex-disjoint paths or a single cycle in the complete graph with vertex set $V(\textsc{Pe}^*(b))$ such that:

- $\mathcal{P}$ consists of exactly one path P_e between u and v for every $e = \{u, v\} \in M$ or $\mathcal{P}$ is a cycle and $M = \emptyset$.
- $\{\, \mathrm{IN}(P) \mid P \in \mathcal{P} \,\}$ is a partition of $(V(\textsc{Pe}(b)) \setminus \{s, t\}) \cup S$, where $\mathrm{IN}(P)$ denotes the set of inner vertices of P.
- there is a planar drawing $D(b, X)$ of $\textsc{Pe}^*(b) \cup \bigcup_{P \in \mathcal{P}} P$ with outer-face f such that $\sigma(f) = \{s, r, r', t, l', l\}$.

The way we define the types $X = (\psi, M, S)$ of a node b allows us to associate each witness $W = (D, D_H, G_H, H)$ with a type, denoted by $\Gamma_W(b)$, based on the restriction of the witness to the respective pertinent graph.

4.3 Framework for Sphere-cut Decomposition

Here, we introduce our framework to simplify the computation of records via bottom-up dynamic programming along a sphere-cut decomposition. Since the framework is independent of the type of records one aims to compute, we believe that the framework is widely applicable and therefore interesting in its own right. In particular, we introduce a simplified framework for computing the types of arcs (or, equivalently, nooses) in sphere-cut decompositions.

Indeed, the central ingredient of any dynamic programming algorithm on sphere-cut decompositions is a procedure that given an inner node with parent arc a_P and child arcs a_L and a_R computes the set of types for the noose O_{a_P} from the set of types for the nooses O_{a_L} and O_{a_R}. Unfortunately, there is no simple way to obtain O_{a_P} from O_{a_L} and O_{a_R} and this is why computing the set of types for O_{a_P} from the set of types for O_{a_L} and O_{a_R} usually involves a technical and cumbersome case distinction [18]. To circumvent this issue, we

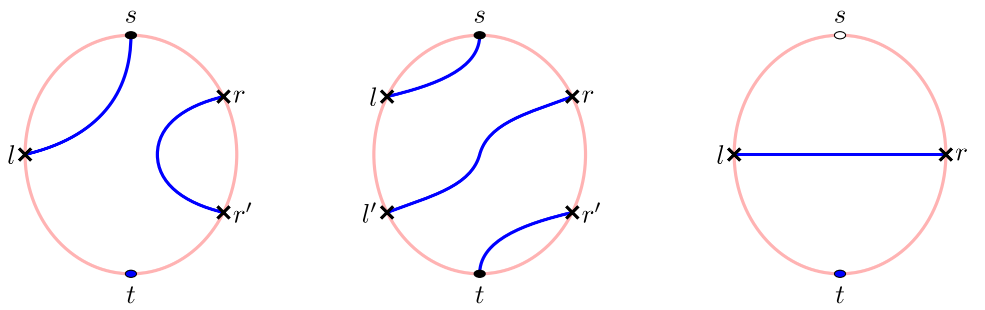

Figure 4 The figure shows three different types of a node in an SPQR-tree with reference edge (s, t), i.e., the types shown are (from left to right): $(\{\{L \to \{l\}\}, \{R \to \{r, r'\}\}, \{\{l, s\}, \{r, r'\}\}, \{t\})$, $(\{\{L \to \{l, l'\}\}, \{R \to \{r, r'\}\}, \{\{l, s\}, \{l', r\}, \{t, r'\}\}, \emptyset)$, and $(\{\{L \to \{l\}\}, \{R \to \{r\}\}, \{\{l, r\}\}, \{t\})$. The subset of $\{l, l'\}$ and $\{r, r'\}$ that appears corresponds to $\psi(L)$ and $\psi(R)$ respectively. The blue edges correspond to the matching M and the blue vertices corresponds to S.

introduce a simple operation, i.e., the $\oplus$ (**XOR**) operation defined below, and show that the noose O_{a_p} can be obtained from the nooses O_{a_L} and O_{a_R} using merely a short sequence – one of length at most four – of $\oplus$ operations.

Central to our framework is the notion of *weak nooses*, which are defined below and can be seen as intermediate results in the above-mentioned sequence of simple operations from the child nooses to the parent noose; in particular, weak nooses are made up of subcurves of the nooses in the sphere-cut decomposition. Let G be a biconnected multi-graph and let $\mathcal{B}$ be an SPQR-tree of G. Let b be an R-node or S-node of $\mathcal{B}$ with pertinent graph $\textsc{Pe}(b)$. Let $\langle T_b, \lambda_b, \Pi_b\rangle$ be a sphere-cut decomposition of $\textsc{Sk}(b)$ and a be an arc of T_b with pertinent graph $\textsc{Pe}(b, a)$. Let $C(T_b)$ be the set of all subcurves of all nooses occurring in T_b, i.e., $C(T_b) = \bigcup_{a \in E(T_b)} O_a$ where O_a is seen as a set of subcurves. We say O is a *weak noose* if O is a noose consisting only of subcurves from $C(T_b)$. For each $O \subseteq C(T_b)$, let $V(O)$ be equal to the vertices of G touched by the noose O.

Having defined weak nooses, we will now define our simplified operation. Let $A \oplus B$ be an exclusive or for two sets A and B, i.e. $A \oplus B = (A \cup B) \setminus (A \cap B)$. We will apply the $\oplus$-operation to weak nooses, whose $\oplus$ is again a weak noose. The following lemma, whose setting is illustrated in Figure 5, is central to our framework as it shows that we can always obtain the noose for the parent arc a_P from the nooses of the child arcs a_L and a_R using a short sequence of $\oplus$-operations such that every intermediate result is a weak noose.

▶ **Lemma 6.** *Let a_P be a parent arc with two child arcs a_L and a_R in a sphere-cut decomposition $\langle T, \lambda, \Pi\rangle$ of a biconnected multi-graph G with the drawing D. There exists a sequence Q of at most 3 $\oplus$-operations such that:*

- *each step generates a weak noose O with $|O| \leq 1 + \max\{|mid(a_P)|, |mid(a_L)|, |mid(a_R)|\}$ as the $\oplus$-operation of two weak nooses O_1 and O_2, whose inside region contains all subcurves in $(O_1 \cap O_2)$,*
- *the last step generates the noose O_{a_P},*
- *Q contains O_{a_L} and O_{a_R} and at most two new weak nooses, each of them bounds the edge-less graph of size 3.*

We are now ready to define the types of weak nooses, which informally can be seen as a generalization of the types of nodes in an SPQR-tree introduced in Subsection 4.2. An illustration of the types is also provided in Figure 7. In the following we fix an arbitrary order π_G of the vertices in G. A type of a weak noose O is a triple (ψ, M, S) such that:

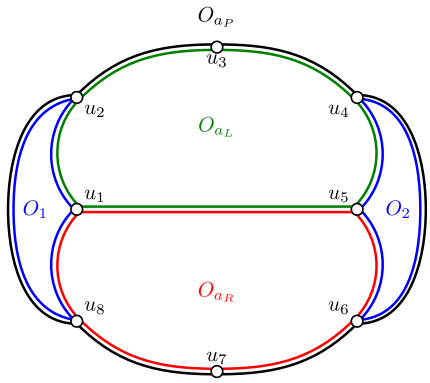

Figure 5 An illustration of the relationship of the parent noose O_{a_P} and the child nooses O_{a_L} and O_{a_R}. The illustration represents the case of Lemma 6 where $O' = O_{a_P} \oplus O_{a_L} \oplus O_{a_R}$ consists of two disjoint weak nooses (triangles) O_1 and O_2.

(1) ψ is a function that for each subcurve $c = (\{u,v\}, f)$ in O, i.e., the subcurve of O between u and v in face f, returns a sequence of at most two new nodes, (2) S is a subset of $V(O)$, and (3) $M \subseteq \{\,\{u,v\} \mid u,v \in V(\psi) \cup (V(O) \setminus S) \wedge u \neq v\,\}$, $V(\psi) \subseteq V(M)$, and M is a non-crossing matching w.r.t. the circular order $\pi^\circ(\psi)$ defined as follows. $\pi^\circ(\psi)$ is the circular order obtained from the circular order $\pi^\circ(O)$ of $V(O)$ after adding $\psi(c)$ between u and v, for every $c = (\{u,v\}, f) \in O$ assuming that $\pi_G(u) < \pi_G(v)$.

The semantics for the types as well as the definition of a type given a witness are now defined in a similar way as in the case of types for SPQR-tree nodes.

5 An FPT-algorithm for SUBHAM using Treewidth

In this section we show that SUBHAM admits a constructive single-exponential fixed-parameter algorithm parameterized by treewidth.

▶ **Theorem 7.** SUBHAM *can be solved in time* $2^{\mathcal{O}(tw)} \cdot n^{\mathcal{O}(1)}$, *where tw is the treewidth of the input graph.*

Since the treewidth of an n-vertex planar graph is upper-bounded by $\mathcal{O}(\sqrt{n})$ [28, 39, 44] and there are single-exponential constant-factor approximation algorithms for treewidth [37], Theorem 7 immediately implies the following corollary.

▶ **Corollary 8.** SUBHAM *can be solved in time* $2^{\mathcal{O}(\sqrt{n})}$.

The main component used towards proving Theorem 7 is the following lemma, from which Theorem 7 follows as an easy consequence .

▶ **Lemma 9.** *Let G be a biconnected multi-graph with n vertices and m edges and SPQR-tree $\mathcal{B}$. Then, we can decide in time $\mathcal{O}(315^\omega n + n^3)$ whether G is subhamiltonian, where ω is the maximum branchwidth of $\textsc{Sk}(b)$ over all R-nodes and S-nodes b of $\mathcal{B}$.*

The remainder of this section is therefore devoted to a proof of Lemma 9, which we show by providing a bottom-up dynamic programming algorithm along the SPQR-tree of the graph. That is, let G be a biconnected multi-graph, $\mathcal{B}$ be an SPQR-tree of G with associated set $\mathcal{T}$ of sphere-cut decompositions for every R-node and S-node of $\mathcal{B}$. Using a dynamic programming algorithm starting at the leaves of $\mathcal{B}$, we will compute a set $\mathcal{R}(b)$ of all types X satisfying the following two conditions:

(R1) If $X \in \mathcal{R}(b)$, then b has type X.
(R2) If there is a witness $W = (D, D_H, G_H, H)$ for G that *respects* $\mathcal{B}$ such that b has type $X = \Gamma_W(b)$, then $X \in \mathcal{R}(b)$.

Interestingly, we do not know whether it is possible to compute the set of all types X such that b has type X as one would usually expect to be able to do when looking at similar algorithms based on dynamic programming. That is, we do not know whether one can compute the set of types that also satisfies the reverse direction of **(R1)**. While we do not know, we suspect that this is not the case because b might have a type that can only be achieved by crossing some sub-curves of nooses inside of $\textsc{Pe}(b)$ more than twice. Indeed Lemma 3, which allows us to avoid more than two crossings per sub-curve, requires the property that the type of b can be extended to a Hamiltonian cycle of the whole graph, which is clearly not necessarily the case for every possible type of b.

5.1 Handling P-nodes

In this part, we show how to compute the set of types for any P-node in the given SPQR-tree by establishing the following lemma.

▶ **Lemma 10.** *Let b be a P-node of $\mathcal{B}$ such that $\mathcal{R}(c)$ has already been computed for every child c of b in $\mathcal{B}$. Then, we can compute $\mathcal{R}(b)$ in time $\mathcal{O}(\ell)$, where ℓ is the number of children of b in $\mathcal{B}$.*

In the following, let b be a P-node of $\mathcal{B}$ with reference edge (s, t) and let C with $|C| = \ell$ be the set of all children of b in $\mathcal{B}$. Informally, $\mathcal{R}(b)$ is the set of types X such that there is an ordering $\rho = (c_1, \ldots, c_\ell)$ of the children in C and an assignment $\tau : C \to \mathcal{X}$ of children to types with $\tau(c) \in \mathcal{R}(c)$ for every child $c \in C$ that "realizes" the type X for b. The main challenge is to compute $\mathcal{R}(b)$ efficiently, i.e., without having to enumerate all possible orderings ρ and assignments τ. Below, we make this intuition more precise before proceeding.

For a type $X = (\psi, M, S)$ of b and $A \in \{L, R\}$, we let $\#_A(X) = |\psi(A)|$. Moreover, for every $A \in \{s, t\}$, we set $\#_A(X)$ to be equal to 2 if $A \in S$, equal to 1 if $A \in V(M)$ and equal to 0 otherwise. Next, let $\rho = (X_1, \ldots, X_\ell)$ be a sequence of types, where $X_i = (\psi_i, M_i, S_i)$ for every i with $1 \le i \le \ell$. We say that ρ is *weakly compatible* if the following holds:
(C1) for every i with $1 \le i < \ell$, $\#_R(X_i) = \#_L(X_{i+1})$, and
(C2) $\sum_{i=1}^{\ell} \#_s(X_i) \le 2$ and $\sum_{i=1}^{\ell} \#_t(X_i) \le 2$.

Note that **(C1)** corresponds to our assumption made in Lemma 4 that we can add the nooses N_b to any planar drawing D of G such that every face of D contains at most one subcurve of any N_b. This in particular means that if $\textsc{Pe}(c)$ is drawn immediately to the left of $\textsc{Pe}(c')$ for two children c and c' of b, then the subcurves R_c and $L_{c'}$ are identical. Please also refer to Figure 6 for an illustration of these subcurves.

Let ρ be weakly compatible. We define the following auxiliary graph $H(\rho)$. $H(\rho)$ has two vertices s and t and additionally for every i with $1 \le i \le \ell$ and every vertex $v \in V(\psi)$, $H(\rho)$ has a vertex v_i. For convenience, we also use s_i and t_i to refer to s and t, respectively. Moreover, $H(\rho)$ has the following edges:

- for every $1 \le i \le \ell$ if $M_i = \emptyset$ and $S_i = \{s_i, t_i\}$, $H(\rho)$ has a cycle on s_i and t_i,
- for every $1 \le i \le \ell$ if $M_i \neq \emptyset$ then for every $e = \{u, v\} \in M_i$, $H(\rho)$ has the edge $\{u_i v_i\}$,
- for every $1 \le i < \ell$, $H(\rho)$ contains the edge $\{r_i, l_{i+1}\}$ if $r \in \psi_i(R)$ and $l \in \psi_{i+1}(L)$,
- for every $1 \le i < \ell$, $H(\rho)$ contains the edge $\{r'_i, l'_{i+1}\}$ if $r' \in \psi_i(R)$ and $l' \in \psi_{i+1}(L)$.

We say that ρ is *compatible* if it is weakly compatible and furthermore either $H(\rho)$ is acyclic, or $H(\rho) - (\bigcup_{i=1}^{\ell} S_i)$ is a single (Hamiltonian) cycle.

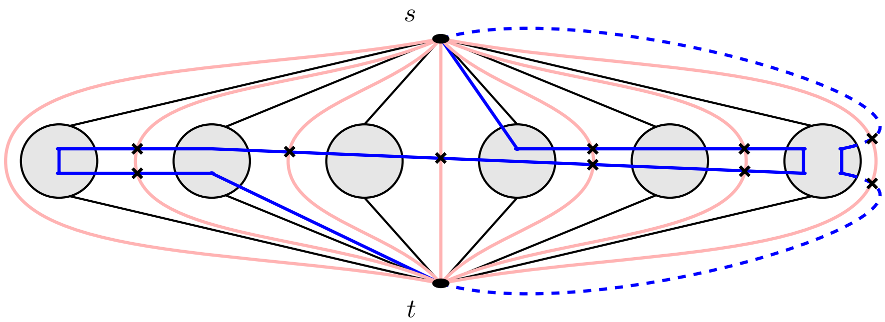

Figure 6 An illustration of how a Hamiltonian Cycle in normal form can interact with a drawing of $\text{PE}(b)$ for a P-node b. Here, the pertinent graphs $\text{PE}(c)$ for all children c of b (without the nodes s and t of the common reference edge (s,t)) are represented by gray ellipses. The Hamiltonian cycle is given in blue with dashed segments representing path segments outside of $\text{PE}(b)$. The red curves represent the subcurves of N_c for every child c of b. In this figure all but the types of the second and fourth pertinent graph are clean. Moreover, the type of the third and fifth pertinent graphs are 1-good and 2-good, respectively, and the types of all other pertinent graphs are bad.

In the following let $\rho = (X_1, \ldots, X_\ell)$ be compatible. We now define the type X associated with ρ, which we denote by $X(\rho)$, as follows. If $H(\rho)$ is a single cycle and $\{s,t\} \subseteq \bigcup_{i=1}^{\ell} S_i$, then we set $X(\rho) = (\psi, \emptyset, \{s,t\})$, where $\psi(L) = \psi(R) = \emptyset$. Otherwise, let $\mathcal{P}(\rho)$ be the set of paths in $H(\rho)$, which can be shown to have their endpoints in $\{s, t, l_1, l_1', r_\ell, r_\ell'\}$. Then, we set $X(\rho) = (\psi, M, S)$, where ψ, M, and S are defined as follows. M contains the set $\{u,v\}$ for every path in $\mathcal{P}(\rho)$ with endpoints u and v; for brevity, we denote l_1, l_1', r_ℓ, r_ℓ' as l, l', r, r', respectively. Moreover, $\psi(L) = V(M) \cap \{l, l'\}$, $\psi(R) = V(M) \cap \{r, r'\}$, and S contains s (t) if $\sum_{i=1}^{\ell} \#_s(X_i) = 2$ ($\sum_{i=1}^{\ell} \#_t(X_i) = 2$).

We say that ρ is *realizable* if there is an ordering $\pi = (c_1, \ldots, c_\ell)$ of the children in C and an assignment $\tau : C \to \mathcal{X}$ from children to types with $\tau(c) \in \mathcal{R}(c)$ for every $c \in C$ such that $\rho = \tau(\pi) = (\tau(c_1), \ldots, \tau(c_\ell))$. The following lemma now allows us to focus on finding the set of all types X for which there is a compatible and realizable ρ such that $X = X(\rho)$.

▶ **Lemma 11.** *The set R containing every type $X \in \mathcal{X}$ such that there is a compatible and realizable ρ with $X = X(\rho)$ satisfies the properties* **(R1)** *and* **(R2)***.*

We will now show that this can be achieved very efficiently because only a constant number, i.e., at most 8 types (and their ordering) need to be specified in order to infer the type of a sequence ρ. Let $X = (\psi, M, S) \in \mathcal{X}$ be a type. We say that X is *dirty* if $\#_s(X) + \#_t(X) > 0$ and otherwise we say that X is *clean*. We say that X is 0-*good*, 1-*good*, and 2-*good*, if X is clean and additionally $M = \emptyset$, $M = \{\{l,r\}\}$, and $M = \{\{l,r\},\{l',r'\}\}$, respectively. We say that X is *good* if it is x-good for some $x \in \{0,1,2\}$ and otherwise we say that X is *bad*. We denote by $\mathcal{X}_G$ and $\mathcal{X}_B$ the subset of $\mathcal{X}$ consisting only of the good respectively bad types. An illustration of these notions is provided in Figure 6.

▶ **Lemma 12.** *Let $\rho = (X_1, \ldots, X_\ell)$ be compatible, then ρ contains at most 8 bad types.*

Next, we will show that any compatible sequence contains at most 8 bad types and that the type $X(\rho)$ is already determined by looking only at the sequence of bad types that occur in ρ. This will then allow us to simulate the enumeration of all possible sequences, by enumerating merely all sequences of at most 8 bad types.

We say that a sequence ρ' is an extension of ρ if ρ is a (not necessarily consecutive) sub-sequence of ρ'. We call a compatible sequence ρ (X,i)-*extendable* for some $X \in \mathcal{X}$ and integer i, if there is a compatible extension ρ' of ρ such that ρ' is obtained by adding i elements of type X to ρ and $X(\rho) = X(\rho')$. We call ρ X-*extendable* if ρ is (X, i)-extendable for any integer i. We say that ρ' is an (X, i)-*extension* of ρ if ρ' is a compatible sequence obtained after adding i elements of type X to ρ and $X(\rho) = X(\rho')$.

▶ **Lemma 13.** *Let $\rho = (X_1, \dots, X_\ell)$ with $X_i = (\psi_i, M_i, S_i)$ and $X \in \mathcal{X}_G$. Then, ρ is $(X, 1)$-extendable if and only if ρ is X-extendable. Moreover, deciding whether ρ is $(X, 1)$-extendable and if so computing an (X, i)-extension ρ' of ρ can be achieved in time $\mathcal{O}(\ell + i)$ for every integer i.*

▶ **Lemma 14.** *Let ρ be a compatible sequence and let ρ' be the sub-sequence of ρ consisting only of the bad types in ρ. Then, ρ' is compatible and $X(\rho) = X(\rho')$.*

At this point, we are ready to describe the algorithm we will use to compute $\mathcal{R}(b)$ (and argue its correctness). The algorithm first enumerates all possible compatible sequences ρ of at most 8 bad types, i.e., $\rho = (Y_1, \dots, Y_r)$ with $r \leq 8$ and $Y_i \in \mathcal{X}_B$ for every i. Note that there are at most $(|\mathcal{X}_B| + 1)^8$ (and therefore constantly many) such sequences and those can be enumerated in constant time. Given one such sequence $\rho = (Y_1, \dots, Y_r)$, the algorithm then tests whether the sequence can be realized given the types available for the children in C as follows. It first uses Lemma 13 to test whether ρ allows for adding a 0-good, 1-good or 2-good type in constant time. Let $A_\rho \subseteq \mathcal{X}_G$ be the set of all good types that can be added to ρ and let C_ρ be the subset of C containing all children c such that $A_\rho \cap \mathcal{R}(c) \neq \emptyset$.

Consider the following bipartite graph Q_ρ having one vertex y_i for every i with $1 \leq i \leq r$ representing the type Y_i on one side and one vertex v_c for every $c \in C$ representing the child c on the other side of the bipartition. Moreover, Q_ρ has an edge between y_i and v_c if $Y_i \in \mathcal{R}(c)$. We claim that ρ can be extended to a compatible and realizable sequence if and only if Q_ρ has a matching that saturates $\{y_1, \dots, y_r\} \cup \{\, v_c \mid c \in C \setminus C_\rho \,\}$. This problem can be solved using a simple reduction to the well-known maximum flow problem. The following lemma now establishes the correctness (i.e., the soundness and completeness) of the algorithm.

▶ **Lemma 15.** *Let $X \in \mathcal{X}$. Then, there is a compatible and realizable sequence ρ with $X = X(\rho)$ if and only if there is a compatible sequence $\rho = (Y_1, \dots, Y_r)$ of bad types with $r \leq 8$ with $X = X(\rho)$ such that the bipartite graph H_ρ has a matching that saturates $\{y_1, \dots, y_r\} \cup \{\, v_c \mid c \in C \setminus C_\rho \,\}$.*

5.2 Handling R-nodes and S-nodes

Here, we will show how to compute a set of types satisfying **(R1)** and **(R2)** for every R-node and S-node of $\mathcal{B}$. To achieve this we will again use a dynamic programming algorithm albeit on a sphere-cut decomposition of $\textsc{Sk}(b)$ instead of on the SPQR-tree. The aim of this subsection is therefore to show the following lemma.

▶ **Lemma 16.** *Let b be an R-node or S-node of $\mathcal{B}$ such that $\mathcal{R}(c)$ has already been computed for every child c of b in $\mathcal{B}$. Then, we can compute $\mathcal{R}(b)$ in time $\mathcal{O}((84\sqrt{14})^\omega \omega \ell + \ell^3)$, where ω is the branchwidth of the graph $\textsc{Sk}(b)$ and ℓ is the number of children of b in $\mathcal{B}$.*

In the following, let b be an R-node or S-node of $\mathcal{B}$ with reference edge (s_b, t_b) and let $\langle T_b, \lambda_b, \Pi_b \rangle$ be a sphere-cut decomposition of $\textsc{Sk}(b)$ that is rooted in $r = \lambda_b^{-1}((s_b, t_b))$. For a weak noose $O \subseteq C(T_b)$, let $\mathcal{A}(O)$ be the set of all types of O satisfying the following two natural analogs of **(R1)** and **(R2)**, i.e.:

(RO1) if $X \in \mathcal{A}(O)$, then O has type X, and **(RO2)** if there is a witness (D, D_H, G_H, H) for G that *respects* $\mathcal{B}$ such that $\Gamma_W(b, O) = X$, where $\Gamma_W(b, O)$ is defined analogously to $\Gamma_W(b)$ for the graph $\textsc{Pe}(b, O)$, then $X \in \mathcal{A}(O)$.

Our aim is to compute $\mathcal{A}(O_{a^r})$ for the arc a^r incident to the root r of T_b. This is achieved by computing $\mathcal{A}(O_a)$ for every inner arc a of T_b via a bottom-up dynamic programming algorithm along T_b; after initially calculating $\mathcal{A}(O_a)$ from $\mathcal{R}(c)$ for every leaf-arc a corresponding to the child c of b. Employing our framework introduced in Subsection 4.3, we only have to show how to compute $\mathcal{A}(O_1 \oplus O_2)$ from $\mathcal{A}(O_1)$ and $\mathcal{A}(O_2)$ for any weak nooses O_1 and O_2.

Let O_1 and O_2 be two weak nooses having type $X_1 = (\psi_1, M_1, S_1)$ and type $X_2 = (\psi_2, M_2, S_2)$, respectively. We say that X_1 and X_2 are *compatible* if

(1) $O = O_1 \oplus O_2$ is a weak noose,
(2) the inside region of the noose O contains all subcurves in $(O_1 \cap O_2)$,
(3) $\forall c \in O_1 \cap O_2$, it holds $\psi_1(c) = \psi_2(c)$,
(4) for every $u \in V(O_1 \cap O_2) \setminus V(O_1 \oplus O_2)$, it holds that u is only in one of following sets: S_1, S_2 or $V(M_1) \cap V(M_2)$, and
(5) the multi-graph obtained from the union of M_1 and M_2 is acyclic, or is one cycle and $V(O) \subseteq S_1 \cup S_2 \cup (V(M_1) \cap V(M_2))$,
(6) if X_1 is the full type, then X_2 is the empty type and $V(O_2) \subseteq V(O_1)$, and vice versa.

We denote by $X_1 \circ X_2$ the *combined type* $X = (\psi, M, S)$ of $X_1 = (\psi_1, M_1, S_1)$ and $X_2 = (\psi_2, M_2, S_2)$ for the weak noose $O = O_1 \oplus O_2$ that is defined as follows and also illustrated in Figure 7. For each $c \in O$, if $c \in O_1$ then $\psi(c)$ is equal to $\psi_1(c)$, otherwise $\psi(c)$ is equal to $\psi_2(c)$ and the set S is equal to $(S_1 \cup S_2 \cup (V(M_1) \cap V(M_2))) \cap V(O)$, i.e., any vertex with degree two w.r.t. X must be in $V(O)$ and have degree two already w.r.t. X_1 or X_2, or it must be in both matchings M_1 and M_2. If either X_1 or X_2 is a full type, then by **(6)** we get that $M_1 = M_2 = M = \emptyset$ and $X_1 \circ X_2$ is the full type. If the multi-graph $M_1 \cup M_2$ is one cycle, then by **(5)** we get that $M = \emptyset$ and $X_1 \circ X_2$ is the full type. Otherwise, due to **(5)**, the multi-graph $M_1 \cup M_2$ is acyclic and corresponds to a set of paths. Therefore, the matching M is the set containing the two endpoints for every path in $M_1 \cup M_2$.

▶ **Observation 17.** *Let X_1 and X_2 be two types defined on the weak nooses O_1 and O_2, respectively. Then, we can check whether X_1 and X_2 are compatible and if so compute the type $X_1 \circ X_2$ in time $\mathcal{O}(|O_1| + |O_2|)$.*

To show the correctness of our approach it now remains to show that: (1) if there is a witness W for G that respects $\mathcal{B}$, then for every two weak nooses O_1 and O_2 it holds that $\Gamma_W(b, O_1)$ and $\Gamma_W(b, O_2)$ are compatible types and $\Gamma_W(b, O) = \Gamma_W(b, O_1) \circ \Gamma_W(b, O_2)$ and (2) if O_1 and O_2 have compatible types X_1 and X_2, then $O = O_1 \oplus O_1$ has type $X_1 \circ X_2$.

5.3 Putting Everything Together

Finally, we show how to compute the set of types for every leaf (Q-node) l of $\mathcal{B}$ in time $\mathcal{O}(1)$; informally, since $\textsc{Pe}(b)$ is just an edge (s, t), $\mathcal{R}(l)$ contains all types that do not allow the Hamiltonian cycle to cross from left to right without using either s or t. Together with Lemma 10 and 16, this then concludes the proof of Lemma 9.

6 An Algorithm Using the Feedback Edge Number

In this section, we establish the following theorem:

▶ **Theorem 18.** Book Thickness *is fixed-parameter tractable when parameterized by the feedback edge number of the input graph.*

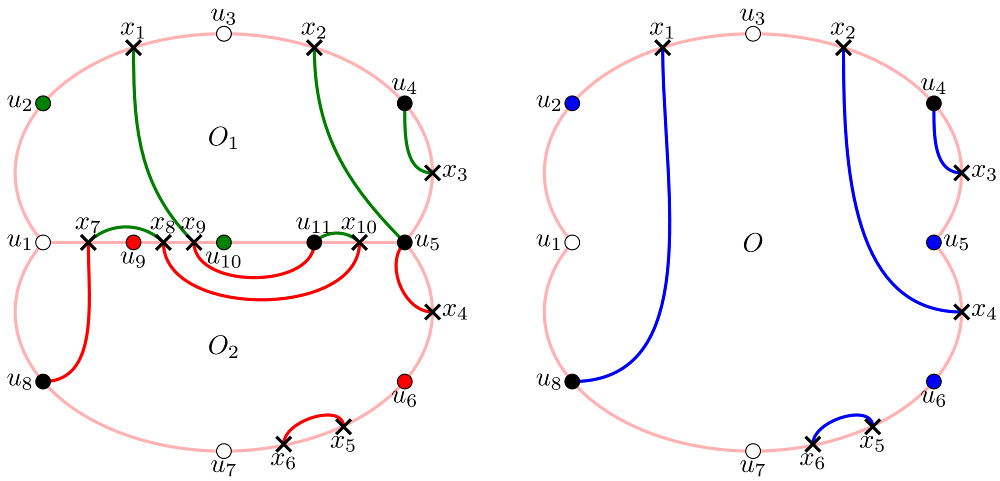

Figure 7 An illustration of combining two compatible types $X_1 = (\psi_1, M_1, S_1)$ and $X_2 = (\psi_2, M_2, S_2)$ for two weak nooses O_1 and O_2 into the combined type $X = (\psi, M, S) = X_1 \circ X_2$ for $O = O_1 \oplus O_2$. Vertices of the graph are represented as circles and vertices subdividing the nooses, i.e., vertices in $V(\psi_1) \cup V(\psi_2)$, are represented as crosses. Black vertices are the vertices that are within a matching, i.e., the vertices in $V(M_1) \cup V(M_2)$, green (red) vertices are the vertices in S_1 (S_2) and all other vertices of the graph are white.

The result is achieved by separately handling two cases: one where the targeted number of pages is greater than 2, or where it is precisely 2. Both cases are handled by a kernelization procedure, and in both cases it is easy to show that pendant vertices can be safely removed. At this point, the target graph consists of a tree plus k edges, whereas the only part that may remain large in this tree are paths of degree-2 vertices. In the former case, we obtain a non-trivial proof that allows us to reduce the maximum length of such a path to length that is bounded by an exponential function of the feedback edge number. In the latter case (which is equivalent to solving SUBHAM), the reduction step is easier and we in fact obtain a linear kernel for the problem:

▶ **Theorem 19.** SUBHAM *parameterized by the feedback edge number k admits a kernel with at most $12k - 8$ vertices and at most $14k - 9$ edges.*

Moreover, by combining Theorem 19 with the subexponential algorithm of Corollary 8, we can slightly strengthen our main result as follows.

▶ **Corollary 20.** SUBHAM *can be solved in time $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$, where k is the feedback edge number of the input graph.*

7 Concluding Remarks

While our main algorithmic result settles the complexity of computing 2-page book embeddings under the exponential time hypothesis, many questions remain when one aims at computing k-page book embeddings for a fixed k greater than 2. To the best of our knowledge, even the existence of a single-exponential algorithm for this problem is open.

In terms of the problem's parameterized complexity, it is natural to ask whether one can obtain a generalization of Theorem 7 for computing k-page book embeddings when $k > 2$. In fact, it is entirely open whether computing, e.g., 4-page book embeddings is even in XP when parameterized by the treewidth. In this sense, our positive result for the feedback edge number can be seen as a natural step on the way towards finally settling the structural boundaries of tractability for computing page-optimal book embeddings.

References

1 Bernardo M. Ábrego, Oswin Aichholzer, Silvia Fernández-Merchant, Pedro Ramos, and Gelasio Salazar. The 2-page crossing number of K_n. *Discrete & Computational Geometry*, 49(4):747–777, 2013. doi:10.1007/S00454-013-9514-0.

2 Patrizio Angelini, Marco Di Bartolomeo, and Giuseppe Di Battista. Implementing a partitioned 2-page book embedding testing algorithm. *Proc. GD 2012*, 7704:79–89, 2012. doi:10.1007/978-3-642-36763-2_8.

3 Michael J. Bannister and David Eppstein. Crossing minimization for 1-page and 2-page drawings of graphs with bounded treewidth. *Journal of Graph Algorithms and Applications*, 22(4):577–606, 2018. doi:10.7155/jgaa.00479.

4 Giuseppe Di Battista and Roberto Tamassia. Incremental planarity testing. *Proc. FOCS 1989*, pages 436–441, 1989. doi:10.1109/SFCS.1989.63515.

5 Michael A. Bekos, Martin Gronemann, and Chrysanthi N. Raftopoulou. Two-page book embeddings of 4-planar graphs. *Algorithmica*, 75(1):158–185, 2016. doi:10.1007/s00453-015-0016-8.

6 Frank Bernhart and Paul C. Kainen. The book thickness of a graph. *Journal of Combinatorial Theory, Series B*, 27(3):320–331, 1979. doi:10.1016/0095-8956(79)90021-2.

7 Sujoy Bhore, Robert Ganian, Fabrizio Montecchiani, and Martin Nöllenburg. Parameterized algorithms for book embedding problems. *Journal of Graph Algorithms and Applications*, 24(4):603–620, 2020. doi:10.7155/jgaa.00526.

8 Daniel Bienstock and Clyde L. Monma. Optimal enclosing regions in planar graphs. *Networks*, 19(1):79–94, 1989. doi:10.1002/NET.3230190107.

9 Daniel Bienstock and Clyde L. Monma. On the complexity of embedding planar graphs to minimize certain distance measures. *Algorithmica*, 5(1):93–109, 1990. doi:10.1007/BF01840379.

10 Hans L. Bodlaender, Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Information and Computation*, 243:86–111, 2015. doi:10.1016/J.IC.2014.12.008.

11 F. Chung, F. Leighton, and A. Rosenberg. Embedding graphs in books: a layout problem with applications to VLSI design. *SIAM Journal on Algebraic Discrete Methods*, 8(1):33–58, 1987. doi:10.1137/0608002.

12 Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

13 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

14 Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Fast hamiltonicity checking via bases of perfect matchings. *Journal of the ACM*, 65(3):12:1–12:46, 2018. doi:10.1145/3148227.

15 Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michal Pilipczuk, Johan M. M. van Rooij, and Jakub Onufry Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. *ACM Transactions on Algorithms*, 18(2):17:1–17:31, 2022. doi:10.1145/3506707.

16 Hubert de Fraysseix, Patrice Ossona de Mendez, and János Pach. A left-first search algorithm for planar graphs. *Discrete & Computational Geometry*, 13:459–468, 1995. doi:10.1007/BF02574056.

17 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

18 Frederic Dorn, Eelko Penninkx, Hans L. Bodlaender, and Fedor V. Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58(3):790–810, 2010. doi:10.1007/S00453-009-9296-1.

19 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

20 Vida Dujmović and David R. Wood. On linear layouts of graphs. *Discrete Mathematics & Theoretical Computer Science*, 6(2):339–358, 2004. doi:10.46298/dmtcs.317.

21 Vida Dujmovic and David R. Wood. Graph treewidth and geometric thickness parameters. *Discrete & Computational Geometry*, 37(4):641–670, 2007. doi:10.1007/s00454-007-1318-7.

22 Toshiki Endo. The pagenumber of toroidal graphs is at most seven. *Discrete Mathematics*, 175(1):87–96, 1997. doi:10.1016/S0012-365X(96)00144-6.

23 Robert Ganian, Fabrizio Montecchiani, Martin Nöllenburg, and Meirav Zehavi. Parameterized complexity in graph drawing (dagstuhl seminar 21293). *Dagstuhl Reports*, 11(6):82–123, 2021. doi:10.1016/j.artint.2017.12.006.

24 Robert Ganian, Haiko Mueller, Sebastian Ordyniak, Giacomo Paesani, and Mateusz Rychlicki. A tight subexponential-time algorithm for two-page book embedding, 2024. arXiv:2404.14087.

25 Joseph L. Ganley and Lenwood S. Heath. The pagenumber of k-trees is $O(k)$. *Discrete Applied Mathematics*, 109(3):215–221, 2001. doi:10.1016/S0166-218X(00)00178-5.

26 M. R. Garey, D. S. Johnson, and R. Endre Tarjan. The planar hamiltonian circuit problem is np-complete. *SIAM Journal on Computing*, 5(4):704–714, 1976. doi:10.1137/0205049.

27 Emilio Di Giacomo and Giuseppe Liotta. The hamiltonian augmentation problem and its applications to graph drawing. *Proc. WALCOM 2010, LNCS*, 5942:35–46, 2010. doi:10.1007/978-3-642-11440-3_4.

28 Qian-Ping Gu and Hisao Tamaki. Improved bounds on the planar branchwidth with respect to the largest grid minor size. *Algorithmica*, 64(3):416–453, 2012. doi:10.1007/S00453-012-9627-5.

29 Carsten Gutwenger, Petra Mutzel, and René Weiskircher. Inserting an edge into a planar graph. *Algorithmica*, 41(4):289–308, 2005. doi:10.1007/S00453-004-1128-8.

30 András Gyárfás and Jenö Lehel. Covering and coloring problems for relatives of intervals. *Discrete Mathematics*, 55(2):167–180, 1985. doi:10.1016/0012-365X(85)90045-7.

31 Christian Haslinger and Peter F. Stadler. RNA structures with pseudo-knots: Graph-theoretical, combinatorial, and statistical properties. *Bulletin of Mathematical Biology*, 61(3):437–467, 1999. doi:10.1006/bulm.1998.0085.

32 Lenwood S. Heath. Embedding outerplanar graphs in small books. *SIAM Journal on Algebraic Discrete Methods*, 8(2):198–218, 1987. doi:10.1137/0608018.

33 Seok-Hee Hong and Hiroshi Nagamochi. Two-page book embedding and clustered graph planarity. Technical report, Citeseer, 2009.

34 Seok-Hee Hong and Hiroshi Nagamochi. Simpler algorithms for testing two-page book embedding of partitioned graphs. *Theoretical Computer Science*, 725:79–98, 2018. doi:10.1016/J.TCS.2015.12.039.

35 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001. doi:10.1006/JCSS.2001.1774.

36 Hugo Jacob and Marcin Pilipczuk. Bounding twin-width for bounded-treewidth graphs, planar graphs, and bipartite graphs. *Proc. WG 2022*, 13453:287–299, 2022. doi:10.1007/978-3-031-15914-5_21.

37 Tuukka Korhonen. A single-exponential time 2-approximation algorithm for treewidth. *Proc. FOCS 2021*, pages 184–192, 2021. doi:10.1109/FOCS52979.2021.00026.

38 Seth M. Malitz. Genus g graphs have pagenumber $O(\sqrt{g})$. *Journal of Algorithms*, 17(1):85–109, 1994. doi:10.1006/jagm.1994.1028.

39 Dániel Marx. Four shorts stories on surprising algorithmic uses of treewidth. *Treewidth, Kernels, and Algorithms*, 12160:129–144, 2020. doi:10.1007/978-3-030-42071-0_10.

40 Dániel Marx, Marcin Pilipczuk, and Michal Pilipczuk. A subexponential parameterized algorithm for directed subset traveling salesman problem on planar graphs. *SIAM Journal on Computing*, 51(2):254–289, 2022. doi:10.1137/19M1304088.

41 Jaroslav Nešetřil and Patrice Ossona de Mendez. *Sparsity – Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-27875-4.

42 Malgorzata Nowicka, Vinay K. Gautam, and Pekka Orponen. Automated rendering of multi-stranded dna complexes with pseudoknots, 2023. arXiv:2308.06392.

43 Neil Robertson and Paul D. Seymour. Graph minors. x. obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52(2):153–190, 1991. doi:10.1016/0095-8956(91)90061-N.

44 Neil Robertson, Paul D. Seymour, and Robin Thomas. Quickly excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 62(2):323–348, 1994. doi:10.1006/JCTB.1994.1073.

45 Johannes Uhlmann and Mathias Weller. Two-layer planarization parameterized by feedback edge set. *Theoretical Computer Science*, 494:99–111, 2013. doi:10.1016/J.TCS.2013.01.029.

46 Avi Wigderson. The complexity of the hamiltonian circuit problem for maximal planar graphs. *Technical Report*, 1982.

47 Mihalis Yannakakis. Embedding planar graphs in four pages. *Journal of Computer and System Sciences*, 38(1):36–67, 1989. doi:10.1016/0022-0000(89)90032-9.