Algorithms for computing the 2-vertex-connected components and the 2-blocks of directed graphs

Dissertation zur Erlangung des akademischen Grades
Doctor rerum naturalium (Dr. rer. nat.)

vorgelegt der Fakultät für Informatik und Automatisierung
der Technischen Universität Ilmenau

von

M.Sc. Raed Jaberi

1. Gutachter: Univ.-Prof. Dr. Martin Dietzfelbinger, TU Ilmenau
2. Gutachter: Jun.-Prof. Dr. Jens M. Schmidt, TU Ilmenau
3. Gutachter: Prof. Giuseppe F. Italiano, Ph.D, Università di Roma

Tag der Einreichung: 06. Juli 2015
Die wissenschaftliche Aussprache (nichtöffentlicher Teil): 15.10.2015
Die wissenschaftliche Aussprache (öffentlicher Teil): 11.11.2015

urn:nbn:de:gbv:ilm1-2015000474

Abstract

In this dissertation we study several problems in directed graphs. First we consider the problem of computing the 2-vertex-connected components of directed graphs. We present a new algorithm for solving this problem in $O(nm)$ time and we consider three applications of this algorithm.

Let $G = (V, E)$ be a directed graph. A 2-*directed block* in G is a maximal vertex set $C^{2d} \subseteq V$ with $|C^{2d}| \geq 2$ such that for each pair of distinct vertices $x, y \in C^{2d}$ there exist two vertex-disjoint paths from x to y and two vertex-disjoint paths from y to x in G. We present two algorithms for computing the 2-directed blocks of G in $O(\min\{m, (t_{sap} + t_{sb})n\}n)$ time, where t_{sap} is the number of strong articulation points of G and t_{sb} is the number of strong bridges of G. Furthermore, we study two related concepts: the 2-strong blocks and the 2-edge blocks of G. We give two algorithms for computing the 2-strong blocks of G in $O(\min\{m, t_{sap}n\}n)$ time and we show that the 2-edge blocks of G can be computed in $O(\min\{m, t_{sb}n\}n)$ time. We also study some optimization problems related to the strong articulation points and the 2-blocks of a directed graph. Given a strongly connected graph $G = (V, E)$, we want to find a minimum strongly connected spanning subgraph $G^* = (V, E^*)$ of G such that the strong articulation points of G coincide with the strong articulation points of G^*. We show that there is a linear time 17/3-approximation algorithm for this NP-hard problem. We also consider the problem of finding a minimum strongly connected spanning subgraph with the same 2-blocks in a strongly connected graph G. We present approximation algorithms for three versions of this problem, depending on the type of 2-blocks.

Unlike 2-vertex-connected components, removing the strong articulation points of G that do not belong to the 2-directed blocks of G may change these blocks. We study the problem of finding a minimum cardinality subset V^* of strong articulation points that are not in these blocks such that the removal of V^* has the same effect. Moreover, we study other versions of this problem depending on the type of 2-blocks.

A directed graph $G = (V, E)$ is called singly-connected if for each pair of vertices $v, w \in V$ there is at most one simple path from v to w in G. Buchsbaum

and Carlisle (1993) gave an algorithm for testing whether G is singly-connected in $O(n^2)$ time. Here we describe a refined version of this algorithm with running time $O(s \cdot t + m)$, where s and t are the number of sources and sinks, respectively, in the reduced graph G^r obtained by first contracting each strongly connected component of G into one vertex and then eliminating vertices of indegree or outdegree 1 by a contraction operation. Moreover, we show that the problem of finding a minimum cardinality edge subset $C \subseteq E$ (respectively, vertex subset $F \subseteq V$) whose removal from G leaves a singly-connected graph is NP-hard.

Zusammenfassung

In dieser Dissertation beschäftigen wir uns mit mehreren Problemen in gerichteten Graphen. Zuerst betrachten wir das Problem der Berechnung der 2-fach knotenzusammenhängenden Komponenten in gerichteten Graphen. Wir präsentieren einen neuen Algorithmus zur Lösung dieses Problems in Zeit $O(nm)$ und wir betrachten drei Anwendungen dieses Algorithmus.

Sei $G = (V, E)$ ein gerichteter Graph. Ein 2-gerichteter Block in G ist eine maximale Knotenmenge $C^{2d} \subseteq V$ mit $|C^{2d}| \geq 2$, so dass für jedes Paar von verschiedenen Knoten $x, y \in C^{2d}$ zwei knoten disjunkte Wege von x nach y und zwei knoten disjunke Wege von y nach x in G existieren. Wir präsentieren zwei Algorithmen zur Berechnung der 2-gerichteten Blöcke von G in Zeit $O(\min\{m, (t_{sap} + t_{sb})n\}n)$, wobei t_{sap} die Anzahl der starken Artikulationspunkte von G und t_{sb} die Anzahl der starken Brücken von G ist. Wir untersuchen zwei weitere relevante Begriffe: die 2-starken Blöcke und die 2-Kanten-Blöcke von G. Wir geben zwei Algorithmen zur Berechnung der 2-starken Blöcke von G in Zeit $O(\min\{m, t_{sap}n\}n)$ an und wir zeigen, dass die 2-Kanten-Blöcke von G in Zeit $O(\min\{m, t_{sb}n\}n)$ bestimmt werden können. Wir untersuchen auch einige Optimierungsprobleme, die mit starken Artikulationspunkten und 2-Blöcken zusammenhängen. Gegeben sei ein stark zusammenhängender Graph $G = (V, E)$. Wir wollen einen minimalen stark zusammenhängenden spannenden Teilgraphen $G^* = (V, E^*)$ von G finden, so dass die starken Artikulationspunkte von G mit den starken Artikulationspunkten von G^* übereinstimmen. Wir zeigen, dass es einen 17/3-Approximationsalgorithmus für dieses NP-schwere Problem gibt, der lineare Laufzeit hat. Wir betrachten auch das Problem der Berechnung eines minimalen stark zusammenhängenden spannenden Teilgraphen mit denselben 2-Blöcken in einem stark zusammenhängenden Graphen. Wir präsentieren Approximationsalgorithmen für drei Versionen dieses Problems, die sich in der Art der 2-Blöcke unterscheiden.

Im Gegensatz zu 2-stark-zusammenhängenden Komponenten kann Entfernung der nicht zu 2-gerichteten Blöcken gehörenden starken Artikulationspunkte aus G diese Blöcke ändern. Wir untersuchen das Problem der Berechnung einer minimalen Teilmenge V^* der starken Artikulationspunkte, die nicht

in diesen Blöcke liegen, so dass das Entfernen von V^* denselben Effekt hat. Außerdem untersuchen wir andere Versionen dieses Problems, die sich in der Art der 2-Blöcke unterscheiden.

Ein gerichteter Graph $G = (V, E)$ heißt einfach zusammenhängend, wenn für jedes Paar von Knoten $v, w \in V$ höchstens ein einfacher Weg von v nach w in G existiert. Buchsbaum und Carlisle (1993) gaben einen Algorithmus an, der in Zeit $O(n^2)$ überprüft, ob G einfach zusammenhängend ist. In dieser Dissertation beschreiben wir eine verbesserte Version dieses Algorithmus, die Laufzeit $O(s \cdot t + m)$ hat, wobei s, t die Anzahl der Quellen beziehungsweise Senken im reduzierten Graphen sind, den man auf folgende Weise erhält: Kontrahieren jeder stark zusammenhängenden Komponente zu einem Knoten und dann Eliminierung von Knoten mit Eingangsgrad oder Ausgangsgrad 1 durch eine Kontrahierenoperation. Außerdem zeigen wir, dass das Problem der Berechnung einer Kantenteilmenge $C \subseteq E$ (beziehungsweise Knotenteilmenge $F \subseteq V$), deren Löschen einen einfach-zusammenhängenden Graphen übrig lässt, NP-schwer ist.

Acknowledgements

Foremost, I would like to thank my supervisor Martin Dietzfelbinger for his support, for reading carefully my results and making valuable comments on them, for giving me a lot of good advice that improved the way of presenting my proofs, and for his patience in listening and discussing all my ideas.

It is a great honor for me that my dissertation was reviewed by Martin Dietzfelbinger, Jens M. Schmidt, and Giuseppe F. Italiano. I thank them for writing reviews on my dissertation.

I would like to thank the present and former staff of the institute of Theoretical Computer Science Dietrich Kuske, Martin Aumüller, Petra Schüller, Christopher Mattern, Jana Kopp, Michael Rink, Sascha Grau, Roy Mennicke, Martin Huschenbett, Faried Abu Zaid, and Olena Prianychnykova for their support and encouragement.

Many thanks to Petra Schüller, Martin Aumüller and Sascha Grau for showing me Goethe's Home and Goethe National Museum in Weimar.

I am grateful to Petra, Jana, Michael, and my colleagues for preparing a wonderful celebration after my defense.

I enjoyed playing board games with Martin Aumüller, Sascha, Michael, Roy, Martin Huschenbett, Adrian, Ralf, Faried, and Olena. I thank them for organizing board games weekly.

And finally I would like to thank my father, my mother, and my brothers Ramez and Raji for their support.

Contents

1. Introduction

Let $G = (V, E)$ be a directed graph with $|V| = n$ vertices and $|E| = m$ edges. The graph G is *strongly connected* if for each pair of vertices $v, w \in V$ there is a path from v to w and a path from w to v in G. A *strongly connected component* (SCC) of G is a maximal vertex set $C \subseteq V$ such that the induced subgraph on C is strongly connected. The strongly connected components (SCCs) of G are disjoint (see, e.g., [Cor+09]). This is illustrated in Figure 1.1.

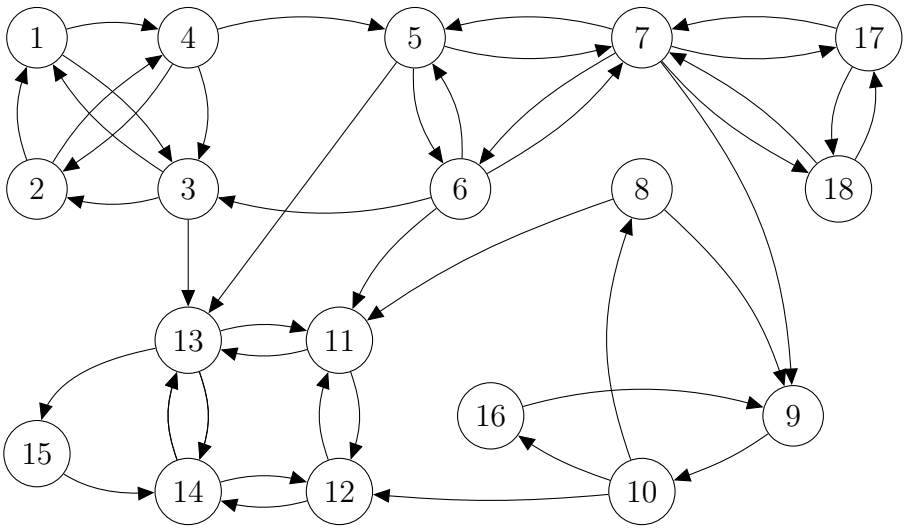


Figure 1.1.: A directed graph G with its SCCs $C_1 = \{1, 2, 3, 4, 5, 6, 7, 17, 18\}$, $C_2 = \{8, 9, 10, 16\}$, and $C_3 = \{11, 12, 13, 14, 15\}$. The 2-vccs of G are $C_1^{2vc} = \{1, 2, 3, 4\}$, $C_2^{2vc} = \{5, 6, 7\}$, $C_3^{2vc} = \{11, 12, 13, 14\}$, $C_4^{2vc} = \{7, 17, 18\}$. Note that while the SCCs are disjoint, the 2-vccs are not necessarily disjoint, like C_2^{2vc}, C_4^{2vc}.

A *strong articulation point* (SAP) of G is a vertex whose removal increases the number of SCCs of G. A *strong bridge* of G is an edge whose removal increases the number of SCCs of G. We use t_{sap} to denote the number of the strong articulation points (SAPs) of G and t_{sb} to denote the number of

the strong bridges of G. A directed graph $G = (V, E)$ is said to be *k-vertex-connected* if it has at least $k+1$ vertices and the induced subgraph on $V \setminus X$ is strongly connected for every $X \subsetneq V$ with $|X| < k$. Thus, a strongly connected graph $G = (V, E)$ is 2-vertex-connected if and only if it has at least 3 vertices and it contains no strong articulation points (SAPs). A 2-vertex-connected component (2-vcc) of a strongly connected graph $G = (V, E)$ is a maximal vertex subset $C^{2vc} \subseteq V$ such that the induced subgraph on C^{2vc} is 2-vertex-connected (see Figure 1.1). The concept was defined in [ES80]. For more information see [ILS12]. In Chapter 2 we study the problem of computing the 2-vertex-connected components (2-vccs) of G. A strongly connected graph G is called 2-*edge connected* if it contains no strong bridges. A 2-*edge-connected component* of G is a maximal vertex subset $C^{2ec} \subseteq V$ such that the induced subgraph on C^{2ec} is 2-edge-connected.

In 2010, Georgiadis [Geo10] presented a linear time algorithm to test whether a strongly connected graph G is 2-vertex-connected or not. Later, Italiano et al. [ILS12] gave a linear time algorithm for the same problem that is faster in practice than the algorithm of Georgiadis [Geo10]. The algorithm of [ILS12] can find all the SAPs of a directed graph G in linear time. Moreover, it solved an open problem posed by Beldiceanu et al. (2005) [BFL05]. The authors of [ILS12] also gave two linear time algorithms for calculating all the strong bridges of a directed graph G. It is well known that the 2-vertex-connected components of an undirected graph can be calculated in linear time [Tar72]. In [ILS12], Italiano et al. posed as an open problem whether the problem of computing the 2-vccs of a directed graph is solvable in linear time. In 1980, Erusalimskii and Svetlov [ES80] gave an algorithm for finding the 2-vccs of a directed graph, whose running time was not analyzed. In Chapter 2 we show that this algorithm runs in $O(nm^2)$ time. Matula [Mat77] gave a polynomial time algorithm for finding all possible k-components of an undirected graph. In 1988, Makino [Mak88] gave an algorithm for computing all the k-vertex-connected components of a directed graph G in time $O(n \cdot \max\{m + n, f\})$, where f is the time bound for calculating a minimum vertex separator of G. According to Henzinger et al. [HKL15], the k-vertex-connected components of a directed graph can be calculated in $O(mn^2)$ using Makino's algorithm since there exists an $O(mn)$-time algorithm for computing minimum vertex separators [Eve75; Gal80; HRG00; Gab06]. Makino's method can be considered as a refinement and an extension of Matula's method (also see [Mak88]). If the algorithm of Italiano et al. [ILS12] is used in Makino's algorithm [Mak88], then Makino's algorithm [Mak88] is able to find all 2-vccs in $O(nm)$ time. A

recent experimental study (2015) [Lui+15] showed that Makino's algorithm for calculating the 2-vccs in directed graphs is not scalable to large graphs. It also showed that the algorithm of Erusalimskii and Svetlov [ES80] is not practicable when the input is large. In Chapter 2 we present a new algorithm for computing the 2-vccs of a directed graph in $O(nm)$ time. The recent experimental study of Luigi et al. [Lui+15] showed that our new algorithm performs well in practice. Moreover, the authors of [Lui+15] presented a refined version of our algorithm for calculating 2-vccs in directed graphs in $O(nm)$ time. The running time is reduced by factor 2. They also gave an $O(nm)$ algorithm for finding 2-edge-connected components. Very recently, Henzinger et al. [HKL15] gave algorithms for computing the 2-vccs and the 2-edge-connected components of a directed graph in $O(n^2)$ time. Moreover, the authors of [HKL15] presented algorithms for finding the k-vertex-connected components in $O(n^3)$ time and the k-edge-connected components in $O(n^2 \log n)$ in a directed graph for any constant k. We briefly describe the algorithm of Henzinger et al. [HKL15] for finding 2-vccs in Appendix A.

The concept of 2-vccs is not ideal because there are directed graphs in which many vertices are well connected with each other but they lie in distinct 2-vccs or in no 2-vcc. This is illustrated in Figure 1.2.

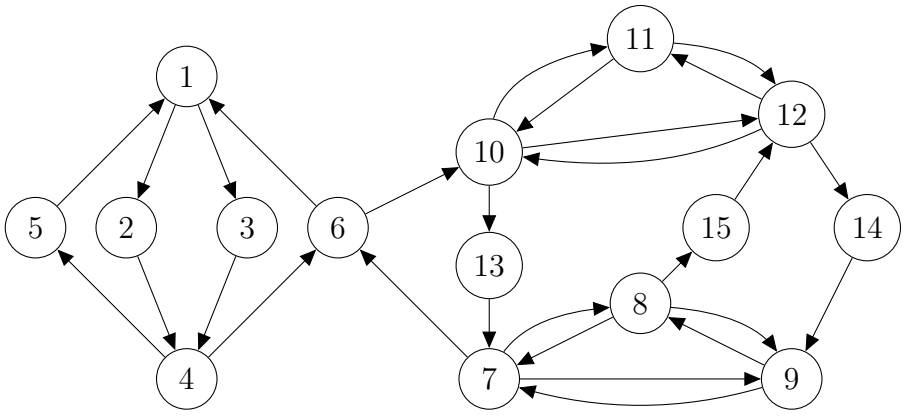


Figure 1.2.: A strongly connected graph G. The 2-vccs of G are $C_1^{2vc} = \{7, 8, 9\}$ and $C_2^{2vc} = \{10, 11, 12\}$. The vertices $8, 11$ lie in distinct 2-vccs of G but there are two vertex-disjoint paths from 8 to 11 and two vertex-disjoint paths from 11 to 8 in G. Notice that the vertices $1, 4$ do not lie in any 2-vcc of G but there exist two vertex-disjoint paths from 1 to 4 and two vertex-disjoint paths from 4 to 1 in G.

In Chapter 3 we study alternative concepts similar to the k-blocks of undirected graphs which were defined in [Car+14] as follows. A *k-block* in an undirected graph $G = (V, E)$ is a maximal vertex set $U \subseteq V$ with $|U| \geq k$ such that no set $X \subseteq V$ with $|X| < k$ separates any two vertices of $U \setminus X$ in the undirected graph G. Carmesin et al. [Car+14] showed that there exists an $O(\min\{k, \sqrt{n}\} \cdot m \cdot n^2)$-time algorithm that calculates all the *k-blocks* in an undirected graph for any fixed k. The 2-blocks in an undirected graph G are similar to the 2-vertex connected components of the undirected graph G, which can be found in linear time using Tarjan's algorithm [Tar72]. In Chapter 3 we introduce and study three new concepts: the 2-directed blocks, the 2-strong blocks, and the 2-edge blocks of directed graphs. A 2-*directed block* in G is a maximal vertex set $C^{2d} \subseteq V$ with $|C^{2d}| \geq 2$ such that for each pair of distinct vertices $x, y \in C^{2d}$, there exist two vertex-disjoint paths from x to y and two vertex-disjoint paths from y to x in G. Since 2-vccs are 2-vertex-connected, they must have at least a linear number of edges. In contrast, the subgraphs induced by the 2-directed blocks may have few or no edges at all , for example the 2-directed block $\{1, 4\}$ in Figure 1.2. Of course, they may also have many edges, like the subgraph induced by the 2-directed block $\{8, 7, 9, 11, 10, 12\}$ in Figure 1.2. A 2-*strong block* in G is a maximal vertex set $C^{2s} \subseteq V$ with $|C^{2s}| \geq 2$ such that for each pair of distinct vertices $x, y \in C^{2s}$ and for each vertex $z \in V \setminus \{x, y\}$, the vertices x and y lie in the same SCC of the graph $G \setminus \{z\}$. A 2-*edge block* in G is a maximal vertex set $C^{2e} \subseteq V$ with $|C^{2e}| \geq 2$ such that for each pair of distinct vertices $x, y \in C^{2e}$, there are two edge-disjoint paths from x to y and two edge-disjoint paths from y to x in G. These concepts capture the idea that it is difficult to separate vertices in a block in slightly different ways, and very different from the concept of 2-vccs. Our new concepts are illustrated in Figure 1.3.

In Chapter 3 we also study some optimization problems related to the SAPs and the 2-blocks of a directed graph. First, we consider the following problem, denoted by MS-SAPs: Given a strongly connected graph $G = (V, E)$, the MS-SAPs problem consists in finding a minimum strongly connected spanning subgraph (MSCSS) $G^* = (V, E^*)$ of G such that the SAPs of G coincide with the SAPs of G^*. Moreover, we consider the problem of finding a MSCSS with the same 2-blocks, defined as follows. Given a strongly connected graph $G = (V, E)$, the goal is to find a subset $E^* \subseteq E$ of minimum size such that $G^* = (V, E^*)$ is strongly connected and the 2-blocks of G coincide with the 2-blocks of $G^* = (V, E^*)$. There are three versions of this problem, depending on the type of 2-blocks: MSCSS with the same 2-directed blocks (denoted by

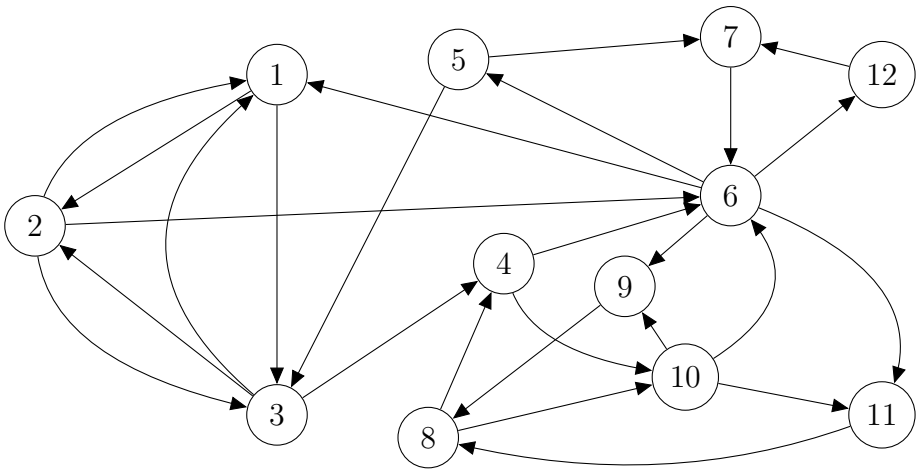


Figure 1.3.: A strongly connected graph G, which contains one 2-vcc $\{1, 2, 3\}$, two 2-directed blocks $\{6, 1, 2, 3\}, \{8, 10, 6, 4\}$, four 2-strong blocks $\{6, 1, 2, 3\}, \{9, 8\}, \{8, 10, 6, 4\}, \{7, 6\}$, and one 2-edge block $\{1, 2, 3, 4, 6, 8, 10\}$. Notice that the 2-vcc $\{1, 2, 3\}$ is a subset of the 2-directed block $\{6, 1, 2, 3\}$. We shall also see in Chapter 3 that each 2-directed block is a subset of a 2-strong block.

MS-2DBs), MSCSS with the same 2-strong blocks (denoted by MS-2SBs), and MSCSS with the same 2-edge blocks (denoted by MS-2EBs). The analogous problems of MS-2DBs and MS-2SBs for undirected graphs can be reduced to the problem of finding a minimum-size 2-vertex-connected spanning subgraph of an undirected graph, which has been studied in [VV00]. Let $G = (V, E)$ be a directed graph. Menger's Theorem for vertex connectivity in directed graphs can be formulated as follows [BR12]. Let x, y be a pair of distinct vertices in G such that $(x, y) \notin E$. Then the maximum number of vertex-disjoint paths from x to y in G is equal to the minimum number of vertices different from x and y whose removal from G destroys all the paths from x to y. This theorem implies that V is a 2-directed block if and only if G is 2-vertex connected. Moreover, by definition, V is a 2-strong block if and only if G is 2-vertex connected. Thus, the problem of finding a minimum-size 2-vertex connected spanning subgraph of a directed graph G is a special case of the problems MS-SAPs, MS-2SBs and MS-2DBs when G is 2-vertex-connected. Menger's Theorem for edge connectivity in directed graphs can be formulated as follows [BR12]. Let v, w be two vertices in G. Then the maximum number of edge-disjoint paths from v

to w in G equals the minimum number of edges whose removal destroys all the paths from v to w. The problem of finding a minimum-cardinality 2-edge connected spanning subgraph of a directed graph $G = (V, E)$ is a special case of the MS-2EBs problem when G is 2-edge-connected since, by Menger's Theorem for edge connectivity, V is a 2-edge block if and only if G is 2-edge connected. Therefore, by results from [GJ79], the problems MS-SAPs, MS-2SBs, MS-2EBs, and MS-2DBs are NP-hard.

Let G be a directed graph. In Chapter 3, we present two algorithms for computing the 2-directed blocks of the graph G in $O(\min\{m, (t_{sap} + t_{sb})n\}n)$ time. We also present two algorithms for computing the 2-strong blocks of G in $O(\min\{m, t_{sap}n\}n)$ time and we show that the 2-edge blocks of G can be computed in $O(\min\{m, t_{sb}n\}n)$ time. Furthermore, we elaborate a linear time 17/3 approximation algorithm for the MS-SAPs problem. We also present a $(2t_{sap}+17/3)$ approximation algorithm for the MS-2SBs problem and a $(2t_{sb}+4)$ approximation algorithm for the MS-2EBs problem. Moreover, we prove that there exists a $(2(t_{sap} + t_{sb}) + 29/3)$ approximation algorithm for the MS-2DBs problem. The content of Chapter 3 was published in [Jab15a].

In independent work, Georgiadis, Italiano, Laura, and Parotsidis [Geo+15a] studied 2-edge blocks and gave linear time algorithms for finding them. This is better than our results in Section 3.3. Recently, the same authors [Geo+15b] gave linear time algorithms for finding 2-directed blocks and 2-strong blocks, improving on our results in Sections 3.1 and 3.2. We briefly describe the algorithm of [Geo+15a] in Appendix A.

Let $G = (V, E)$ be a strongly connected graph. We denote by V^{sap} the set of the SAPs of G and by E^{sb} the set of the strong bridges of G. Let $C_1^{2vc}, C_2^{2vc}, \ldots, C_t^{2vc}$ be the 2-vccs of G. Removing any vertex subset $U \subseteq V \setminus (\bigcup_{1 \leq i \leq t} C^{2vc})$ from the graph G does not have any effect on these 2-vccs. Let $C_1^{2d}, C_2^{2d}, \ldots, C_l^{2d}$ be the 2-directed blocks of G and let x, y be any two distinct vertices in G such that x, y are in the same 2-directed block of G. Note that deleting a subset $U \subseteq V^{sap} \setminus (\bigcup_{1 \leq i \leq l} C_i^{2d})$ may destroy the property that there exist two vertex-disjoint paths from x to y and two vertex-disjoint paths from y to x, as illustrated in Figure 1.4. Unlike 2-vccs, the 2-directed blocks of G may be changed by removing a set of SAPs which do not lie in these 2-directed blocks.

Let $C_1^{2ec}, C_2^{2ec}, \ldots, C_r^{2ec}$ be the 2-edge-connected components of G. Notice that deleting any set F of edges that do not lie in the subgraphs $G[C_1^{2ec}]$, $G[C_2^{2ec}]$,..., $G[C_r^{2ec}]$ has no effect on these 2-edge-connected components since the sets $C_1^{2ec}, C_2^{2ec}, \ldots, C_r^{2ec}$ are the 2-edge-connected components of the directed

graph $G \setminus F$. Let C^{2e} be a 2-edge-block of the graph G and let v, w be two distinct vertices in C^{2e}. The graph obtained by removing a subset of E^{sb} from G may not have the property that there exist two edge-disjoint paths from v to w and two edge-disjoint paths from w to v. Therefore, deleting a subset of E^{sb} may change the 2-edge blocks of G. This is illustrated in Figure 1.5.

In Chapter 4 we study several problems related to the effect of removing SAPs, non-SAPs, and strong bridges on 2-blocks in directed graphs, like the problem of finding a minimum cardinality subset of $V^{sap} \setminus (\bigcup_{1 \leq i \leq l} C_i^{2d})$ whose removal from G leaves a directed graph containing the same 2-directed block as $G \setminus (V^{sap} \setminus (\bigcup_{1 \leq i \leq l} C_i^{2d}))$.

Let $G = (V, E)$ be a directed graph. The graph G is called singly-connected if for each pair of vertices $v, w \in V$ there is at most one simple path from v to w in G. Buchsbaum and Carlisle (1993) gave an algorithm for testing whether G is singly-connected in $O(n^2)$ time. In Chapter 5 we describe a refined version of this algorithm with running time $O(s \cdot t + m)$, where s and t are the number of sources and sinks, respectively, in the reduced graph G^r obtained by first contracting each SCC of G into one vertex and then eliminating vertices of indegree or outdegree 1 by a contraction operation. Moreover, we show that the problem of finding a minimum cardinality edge subset $C \subseteq E$ (respectively, vertex subset $F \subseteq V$) whose removal from G leaves a singly-connected graph is NP-hard. The content of Chapter 5 is joint work with Martin Dietzfelbinger, published in [DJ15].

Graph Terminology and Notation

In this section we recall some basic definitions from [ILS12; LT79; LM69]. A *flowgraph* $G(v) = (V, E, v)$ is a directed graph with $|V| = n$ vertices, $|E| = m$ edges, and a distinguished start vertex $v \in V$ such that for each vertex $w \in V$ there exists a path from v to w. For a flowgraph $G(v) = (V, E, v)$, the dominance relation is defined in the following way: a vertex $u \in V$ is a *dominator* of vertex w in the flowgraph $G(v)$ if each path from v to w includes u. By $dom(w)$ we denote the set of dominators of vertex w. The set of dominators of the start vertex in $G(v)$ is $dom(v) = \{v\}$. For each vertex $w \in V \setminus \{v\}$, the vertices v, w belong to the set $dom(w)$; we call w, v the *trivial dominators* of w. A vertex u is a *non-trivial dominator* in $G(v)$ if there is some $w \notin \{v, u\}$ such that $u \in dom(w) \setminus \{v\}$. The set of all non-trivial dominators in $G(v)$ is denoted by $D(v)$. The dominance relation is transitive and reflexive. A vertex $u \in V$ is an *immediate dominator* of vertex $w \in V \setminus \{v\}$ in $G(v)$ if $u \in dom(w) \setminus \{w\}$

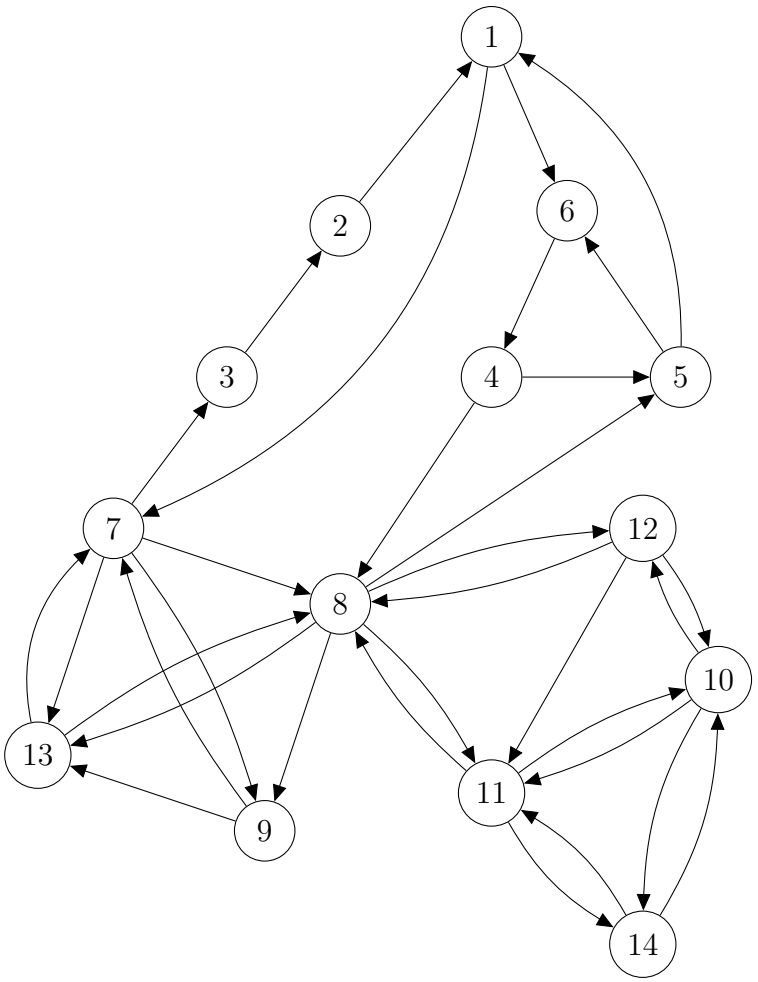


Figure 1.4.: A strongly connected graph $G = (V, E)$. The 2-vccs of G are $C_1^{2v} = \{7, 8, 9, 13\}$ and $C_2^{2vc} = \{8, 10, 11, 12, 14\}$. In this graph there are two 2-directed blocks $C_1^{2d} = \{1, 5, 7, 8, 9, 13\}$ and $C_2^{2d} = \{8, 10, 11, 12, 14\}$. Deleting any vertex subset $U \subseteq V \setminus (C_1^{2v} \cup C_2^{2vc})$ does not have any effect on the 2-vccs C_1^{2v}, C_2^{2vc}. The vertices $2, 3, 4, 6$ are SAPs in G. Additionally, these vertices do not lie in any 2-directed block of G because the vertex 4 has indegree 1 and the vertices $6, 2, 3$ have outdegree 1. In the graph $G \setminus \{2\}$, for any vertex $v \in C_1^{2d} \setminus \{1\}$ there do not exist two vertex-disjoint paths from v to the vertex 1. Therefore, deleting the SAP 2 from G changes the 2-directed block C_1^{2d}. Notice that all the vertices in $C_1^{2d} \setminus C_1^{2v}$ do not belong to any 2-directed block in the graph $G \setminus \{2\}$.

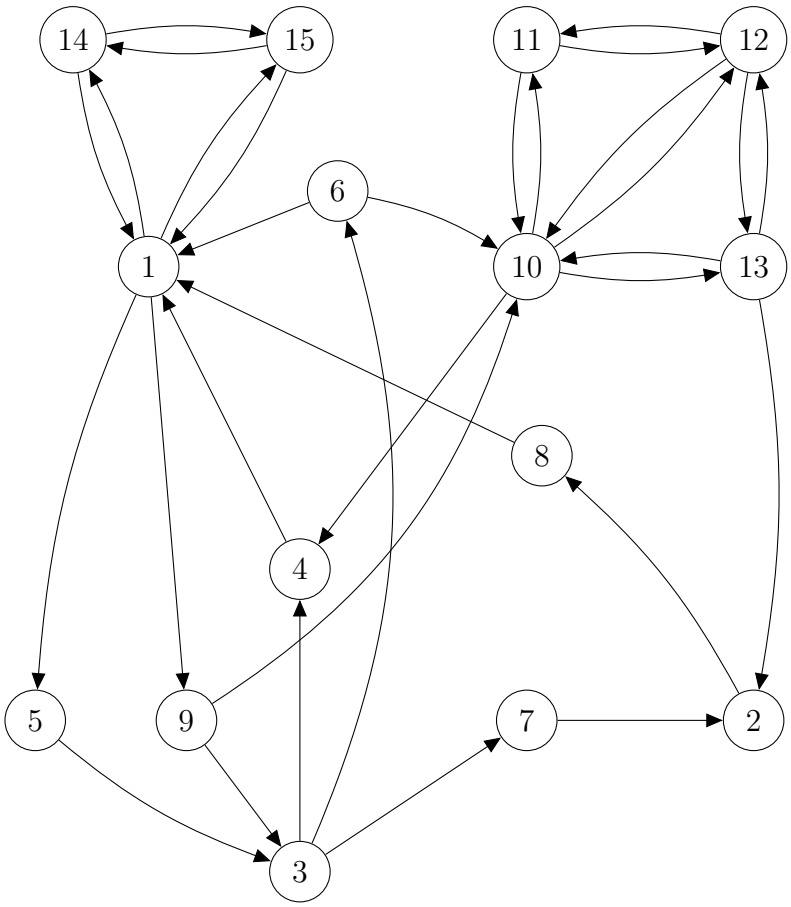


Figure 1.5.: A strongly connected graph G with two 2-edge-connected components $C_1^{2ec} = \{10, 11, 12, 13\}$ and $C_2^{2ec} = \{1, 14, 15\}$. The graph G has only one 2-edge block $C^{2e} = \{3\} \cup C_1^{2ec} \cup C_2^{2ec}$. All vertices in $G \setminus C^{2e}$ have indegree or outdegree 1. Therefore, they do not belong to any 2-edge block of G. Notice that for any vertex $v \in C^{2e} \setminus \{3\}$ there do not exist two edge-disjoint paths from the vertex 3 to v in the graph $G \setminus \{(3,7),(3,6)\}$ since the vertex 3 has outdegree 1 in $G \setminus \{(3,7),(3,6)\}$. Thus, the vertex 3 does not lie in any 2-edge block in $G \setminus \{(3,7),(3,6)\}$. Note that deleting the set $\{(3,7),(3,6)\}$, which is a subset of E^{sb}, from G has a big influence on the 2-edge block C^{2e}, while removing any subset of $E \setminus (C_1^{2ec} \cup C_2^{2ec})$ from G does not affect the 2-edge-connected components C_1^{2ec}, C_2^{2ec}.

and all elements of $dom(w) \setminus \{w\}$ are dominators of u. Each vertex $w \in V \setminus \{v\}$ has a unique immediate dominator, which is denoted by $imd(w)$. The tree formed by the edges (u, w) where u is the immediate dominator of w is called the *dominator tree* of $G(v)$. This tree with root v is denoted by $DT(v)$. Figure 1.6 gives an example of a flowgraph and its dominator tree. Two spanning trees

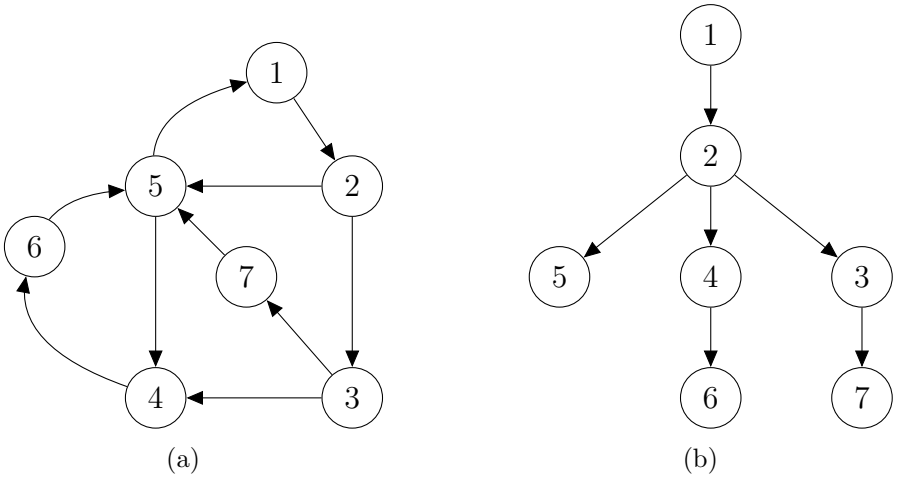
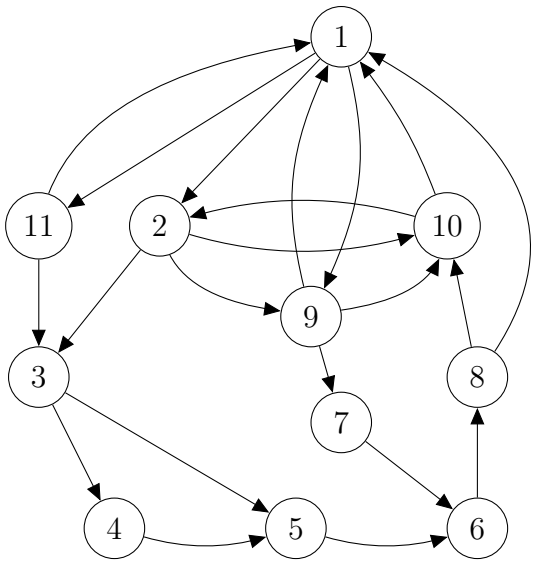


Figure 1.6.: (a) A flowgraph $G(1)$. (b) The dominator tree of $G(1)$.

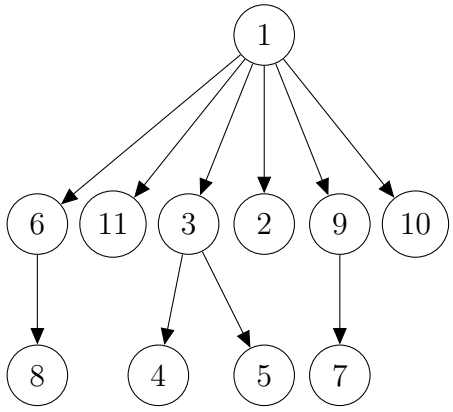
T and T' of $G(v)$ are called independent if for every vertex $w \in V \setminus \{v\}$, the paths from v to w in T and T' contain only $dom(w)$ in common [GT12b]. An edge (x, y) is an *edge dominator* of vertex w if every path from v to w in $G(v)$ contains edge (x, y). Let $G = (V, E)$ be a directed graph. Let $F \subseteq V \times V$ be a set of pairs of vertices. We use $G \setminus F$ to denote the directed graph obtained from G by removing all edges in $E \cap F$ from G. Let U be a subset of V. We use $G \setminus U$ to denote the directed graph obtained from G by removing all the vertices in U and their incident edges. The reversal graph of G is the directed graph $G^R = (V, E^R)$, where $E^R = \{(w, u) \mid (u, w) \in E\}$.

Let $G = (V, E)$ be a strongly connected graph and let v be a vertex in G. Since G^R is strongly connected, $G^R(v) = (V, E^R, v)$ is a flowgraph. By $D^R(v)$ we denote the set of all non-trivial dominators in the flowgraph $G^R(v)$. In [ILS12], Italiano et al. showed that the SAPs of G can be obtained in linear time by computing $D(v) \cup D^R(v)$ and checking whether the graph $G \setminus \{v\}$ is strongly connected. Figure 1.7 demonstrates how their algorithm works. More information on this algorithm can be found in [ILS12; Fir+12].

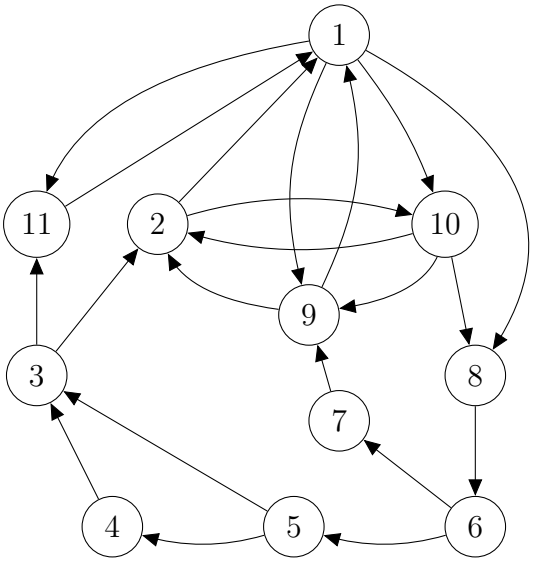
Let $G = (V, E)$ be an undirected graph. A block (2-vertex-connected compo-



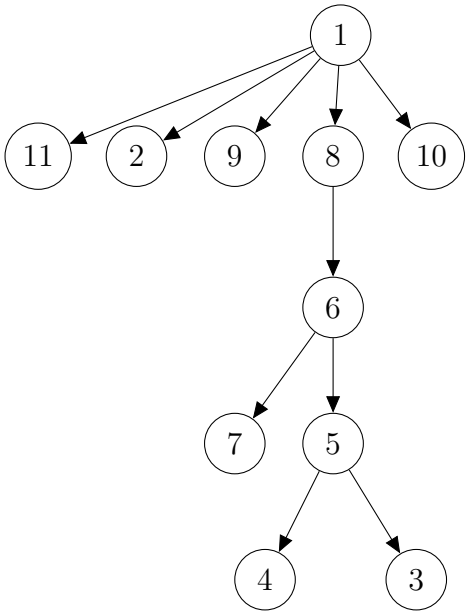
(a) A strongly connected graph G.



(b) The dominator tree $DT(1)$ of the flowgraph $G(1)$, $D(1) = \{3, 6, 9\}$.



(c) The flowgraph $G^R(1)$.



(d) The dominator tree $DT^R(1)$ of $G^R(1)$, $D^R = \{5, 6, 8\}$.

Figure 1.7.: Since the graph $G \setminus \{1\}$ is not strongly connected, the vertex 1 is a SAP in G. Therefore, by [ILS12, Theorem 5.2], $D(1) \cup D^R(1) \cup \{1\} = \{3, 6, 9, 5, 8, 1\}$ is the set of all the SAPs of G.

nent) of G is a maximal connected subgraph of G that contains no articulation points. An undirected graph G is called chordal if every cycle of length at least 4 has a chord [Gav72; RT75].

2. On computing the 2-vertex-connected components of directed graphs

In this chapter we study the problem of computing the 2-vccs of a directed graph. In section 2.1, we briefly describe the algorithm of Erusalimskii and Svetlov [ES80] for computing the 2-vccs of a directed graph and analyze its running time. In section 2.2, we briefly describe an $O(nm)$ algorithm that was implicit already in the work of [Mak88]. In section 2.3, we investigate the relationship between 2-vccs and dominator trees and present our new algorithm for computing all the 2-vccs of a directed graph in $O(nm)$ time. Finally in section 2.4, we consider three applications of our new algorithm, which are approximation algorithms for problems that are generalization of the problem of approximating the smallest 2-vertex-connected spanning subgraph of a 2-vertex-connected directed graph. The content of Chapter 2 was published in [Jab15b].

2.1. The algorithm of Erusalimskii and Svetlov

In this section, we briefly describe the algorithm of Erusalimskii and Svetlov [ES80] for calculating the 2-vccs of a directed graph, and we analyze its running time. The latter analysis was missing in [ES80]. In [ES80], the authors provided an algorithm for computing all *biblocks* of a directed graph, where a biblock of a directed graph $G = (V, E)$ is a maximal vertex set $B \subseteq V$ such that the subgraph induced by B is strongly connected and does not contain any SAP. A biblock is either a 2-vcc, a single vertex or two vertices which are connected by two antiparallel edges. In this chapter we are only interested in computing the 2-vccs of a directed graph. The authors of [ES80] studied a class of directed graphs L defined as follows: A directed graph $G = (V, E)$ belongs to class L if it satisfies the following conditions:

1. If $C_1, C_2, \ldots, C_t$ are the SCCs of G, then there are no edges between C_i and C_j for distinct $i, j \in \{1, \ldots, t\}$.

2. For every SAP v the directed graph $G \setminus \{v\}$ satisfies 1.

Let $G = (V, E)$ be a directed graph. By $U(G)$ we denote the undirected graph formed from G by deleting the directions of the edges. The main idea behind the algorithm of Erusalimskii and Svetlov [ES80] is as follows. Given a directed graph $G = (V, E)$, the algorithm computes a directed graph $G' \in L$ such that the 2-vccs of G coincide with the 2-vccs of G'. Then all 2-vccs of G' can be obtained by calculating the 2-connected components of the undirected graph $U(G')$ since by [ES80, Theorem 6] the 2-vccs of G' coincide with the 2-connected components of $U(G')$. We refer the reader to [Die00; Tar72] for information on the problem of calculating the 2-connected components of an undirected graph. Algorithm 2.1.1 shows the algorithm of Erusalimskii and Svetlov [ES80].

Algorithm 2.1.1
(From [ES80])
Input: A directed graph $G = (V, E)$.
Output: The 2-vccs of G.

```
1   Repeat
2     Compute the SCCs of G.
3     Remove from G the edges between the SCCs of G.
4     for every vertex v ∈ V do
5           Compute the SCCs of G \ {v}.
6           Remove from G the edges between the strongly connected
7           components of G \ {v}.
8   until no edge was removed during step 6.
9   We obtain a directed graph G' ∈ L.
10  Compute the 2-connected components C_1^2vc, C_2^2vc, ..., C_k^2vc of U(G').
12  Output C_1^2vc, C_2^2vc, ..., C_k^2vc.
```

Theorem 2.1.2
The running time of algorithm 2.1.1 is $O(nm^2)$.

Proof. The number of iterations of the repeat-loop is at most m since at least one edge is removed in each iteration. The SCCs of a directed graph can be

found in linear time using Tarjan's algorithm [Tar72]. In each iteration of the repeat-loop, steps 4–7 require $O(n(n+m))$ time. The 2-connected components of an undirected graph can be computed in linear time using Tarjan's algorithm [Tar72]. Thus, the total running time of algorithm 2.1.1 is $O(m(n(m+n))) = O(nm^2)$. □

2.2. Computing the 2-vertex-connected components of directed graphs

In this section, we briefly describe an $O(nm)$ algorithm that was implicit already in the work of [Mak88]. This algorithm is based on the following lemma and the recent results of [ILS12].

Lemma 2.2.1
Let $G = (V, E)$ be a directed graph and let w be a SAP in G. Let C^{2vc} be a 2-vcc of G. Then all vertices of $C^{2vc} \setminus \{w\}$ lie in a SCC C of $G \setminus \{w\}$, i.e. $C^{2vc} \setminus \{w\} \subseteq C$.

Proof. Since C^{2vc} is a 2-vcc of G, the directed graph $G[C^{2vc}]$ does not contain any articulation point. Thus, $G[C^{2vc} \setminus \{w\}]$ is strongly connected. Moreover, $G[C^{2vc} \setminus \{w\}]$ is a subgraph of $G \setminus \{w\}$. Consequently, $C^{2vc} \setminus \{w\}$ is a subset of a SCC of $G \setminus \{w\}$. □

Remark 2.2.2
(a) Reif and Spirakis's separation lemma [RS81] and Makino's separation Lemma [Mak88], of which Lemma 2.2.1 is a special case, is an extension of Matula's separation Lemma [Mat78] for undirected graphs to directed graphs. (b) Matula and Vohra [MV88] presented an algorithm for checking whether a directed graph is k-edge connected. The authors of [MV88] gave an $O(n^3)$ algorithm for computing the k-edge-connected components of a directed graph, which works as follows. The algorithm removes all vertices with indegree at most $k-1$ or outdegree at most $k-1$ and their incident edges. Then it returns the remaining graph if it is k-edge-connected. Otherwise, it finds a minimum edge separator e_c and removes all edges in e_c from the graph and repeats the procedure recursively for each component of the remaining graph. The k-edge-components of a directed graph G are

disjoint [MV88]. Therefore, computing them is easier than calculating the k-vertex-connected components of G. While the algorithm of Matula and Vohra [MV88] deletes minimum edge separators of size at most $k-1$, the algorithm of Makino handles minimum vertex-separators by his separation lemma.

Corollary 2.2.3
Let $G = (V, E)$ be a directed graph and let w be a SAP in G. The 2-vccs of G lie in the subgraphs $G[C_1 \cup \{w\}], G[C_2 \cup \{w\}], \ldots, G[C_t \cup \{w\}]$, where $C_1, C_2, \ldots, C_t$ are the SCCs of $G \setminus \{w\}$.

Corollary 2.2.3 is illustrated in Figure 2.1.

The correctness of Algorithm 2.2.4 follows from Corollary 2.2.3.

Algorithm 2.2.4

Input: A directed graph $G = (V, E)$.
Output: The 2-vccs of G.

```
1  if G is 2-vertex-connected then
2    Output V.
3  else
4    Find a SAP w of G.
5    Compute the SCCs of G \ {w}.
6    for each SCC C of G \ {w} do
7      Recursively compute the 2-vccs of G[C ∪ {w}] and output them.
```

Theorem 2.2.5
Algorithm 2.2.4 runs in $O(nm)$ time.

Proof. The SAPs of a directed graph can be computed in linear time using the algorithm of Italiano et al. [ILS12]. The SCCs of a directed graph can also be computed in linear time using Tarjan's algorithm [Tar72]. Let $C_1, C_2, \ldots, C_t$ be the SCCs of $G \setminus \{w\}$. Clearly, the edge sets of the subgraphs $G[C_1 \cup \{w\}], G[C_2 \cup \{w\}], \ldots, G[C_t \cup \{w\}]$ are disjoint. Thus the total cost at each recursion level is $O(m)$. Since the vertex set of a graph in a recursive call is smaller than the original vertex set, the recursion depth is at most n. Thus the total time is $O(nm)$. □

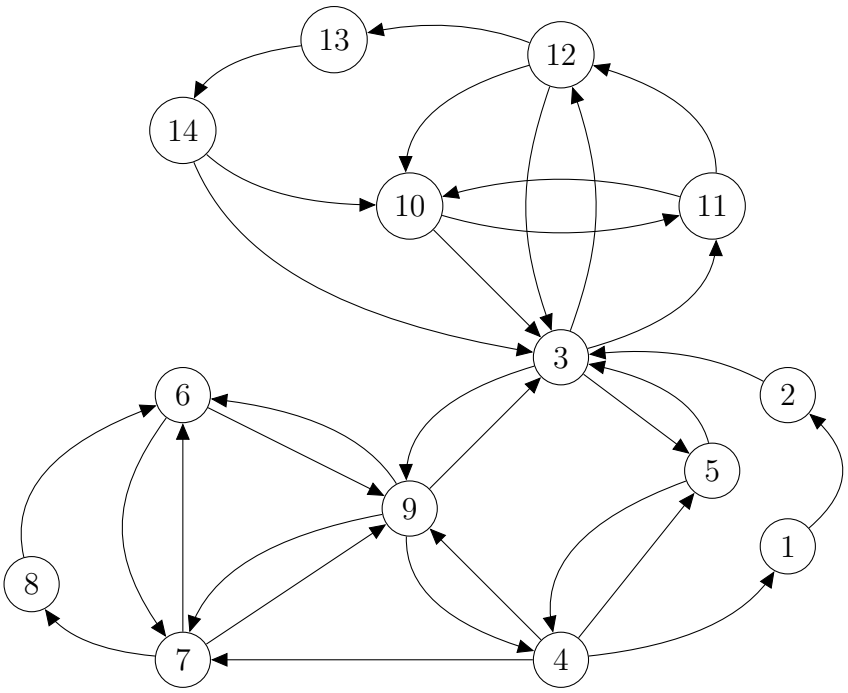


Figure 2.1.: A strongly connected graph G. Its 2-vccs are $C_1^{2vc} = \{3, 4, 5, 9\}$, $C_2^{2vc} = \{6, 7, 9\}$, and $C_3^{2vc} = \{3, 10, 11, 12\}$. The set of the SAPs of G is $\{1, 2, 3, 4, 6, 7, 9, 12, 13, 14\}$. The SCCs of $G \setminus \{9\}$ are $C_1 = \{1, 2, 3, 4, 5, 10, 11, 12, 13, 14\}$ and $C_2 = \{6, 7, 8\}$. Note that the subgraph $G[C_1^{2vc} \cup \{9\}]$ contains the 2-vccs C_1^{2vc} and C_3^{2vc}, and the subgraph $G[C_2 \cup \{9\}]$ includes the 2-vcc C_2^{2vc}.

2.3. The relationship between 2-vertex-connected components and dominator trees

In this section, we prove a connection between the 2-vccs of a directed graph and dominator trees, and we present our new algorithm for computing the 2-vccs of a directed graph.

Theorem 2.3.1
Let $G = (V, E)$ be a strongly connected graph and let v be an arbitrary vertex in G. Let C^{2vc} be a 2-vcc of G. Then either all elements of C^{2vc} are direct successors of some vertex $w \notin C^{2vc}$ or all elements $C^{2vc} \setminus \{w\}$ are

direct successors of some vertex $w \in C^{2vc}$ in the dominator tree $DT(v)$ of the flowgraph $G(v)$.

Proof. We consider two cases:

1. $v \in C^{2vc}$. In this case, all elements of $C^{2vc} \setminus \{v\}$ are direct successors of v in $DT(v)$ since for every vertex x of $C^{2vc} \setminus \{v\}$, there are two vertex-disjoint paths from v to x in $G[C^{2vc}]$, hence in $G(v)$.

2. $v \notin C^{2vc}$. Then there is a vertex $x \in C^{2vc}$ such that $imd(x) = w$ and $w \notin C^{2vc}$. We show, by contradiction, that w is a dominator for every vertex of C^{2vc}. Assume that there is some vertex $y \in C^{2vc}$, $y \neq x$ such that $w \notin dom(y)$. Consequently, there exists a path p from v to y not containing w. This path enters C^{2vc} in vertex $u \in C^{2vc}$. This means that the vertices of p from v to u are outside of C^{2vc}. Moreover, there are two vertex-disjoint paths from u to x in $G[C^{2vc}]$. Thus, there is a path from v to x not containing w. Therefore, we have $w \notin dom(x)$, which contradicts that $imd(x) = w$. Now we consider two cases:

   a) All paths from w to x are completely outside of $G[C^{2vc}]$. Then x is a dominator of all vertices $y \in C^{2vc}$. (Assume that there is a path from v to y avoiding x. By the above, w is on this path. We can extend p inside C^{2vc} to reach x, contradicting the assumption all paths from w to x are completely outside of $G[C^{2vc}]$.)
   For every vertex $y \in C^{2vc}$, x is the immediate dominator of y in $DT(v)$, since there are two vertex-disjoint paths from x to y.

   b) There are at least two vertex-disjoint paths p_1, p_2 from w to x such that p_1 enters C^{2vc} in vertex y and p_2 enters C^{2vc} in vertex y' with $y \neq y'$. Since there are a path from y to y' and a path from y' to y in $G[C^{2vc}]$, there are two vertex-disjoint paths from w to y and two vertex-disjoint paths from w to y' in $G(v)$. Therefore, the vertices y, y' are direct successors of w in $DT(v)$. Now we prove that every vertex $z \in C^{2vc} \setminus \{x, y, y'\}$ is also direct successor of w. There are two case to consider:

      i. All paths from y to z and all paths from y' to z have a vertex $z' \in C^{2vc}$ in common with $z' \notin \{y, y', z\}$. Consequently, all the paths from y to z contain z' in $G[C^{2vc}]$, where $z' \notin \{y, z\}$. Hence, z' is a SAP in $G[C^{2vc}]$ by [ILS12, Lemma 2.1] of Italiano et al., which contradicts that C^{2vc} is 2-vcc of G.

ii. To interrupt all paths from $\{y, y'\}$ to z, one has to remove at least two vertices. We add a vertex $s \notin V$ and two edges $(s, y), (s, y')$ to G. Clearly, s and z are not adjacent. A separator of all paths from s to z is a set of vertices whose removal interrupts all paths from s to z. A minimal separator of all paths from s to z has two vertices. By Menger's Theorem (1927) there are two vertex-disjoint paths from s to z. Thus, there exist a path p from y to z and a path p' from y' to z in $G[C^{2vc}]$ such that p, p' are vertex-disjoint. As a consequence, there are two vertex-disjoint paths from w to z in $G(v)$. Therefore, z is direct successor of w in $DT(v)$. □

By $M(w)$ we denote the set of direct successors of vertex w in the dominator tree of a flowgraph. Algorithm 2.3.2 shows our new algorithm for computing all the 2-vccs of a strongly connected graph G using Theorem 2.3.1.

Algorithm 2.3.2

Input: A strongly connected graph $G = (V, E)$.
Output: The 2-vccs of G.

```
1   if G is 2-vertex-connected then
2     Output V.
3   else
4     Compute the SAPs of G.
5     Choose a vertex v ∈ V that is not a SAP of G.
6     Compute the dominator trees DT(v) and DT^R(v).
7     Choose the dominator tree from {DT(v), DT^R(v)} that contains more
8       non-trivial dominators.
9     for each vertex w ∈ V do
10      if |M(w)| ≥ 2 then
11        if G[M(w) ∪ {w}] is not strongly connected then
12          Compute the SCCs of G[M(w) ∪ {w}].
13          for each SCC C of G[M(w) ∪ {w}] do
14            if |C| ≥ 3 then
15              Recursively compute the 2-vccs of G[C] and output them.
16        else
17          Recursively compute the 2-vccs of G[M(w) ∪ {w}] and output them.
```

Algorithm 2.3.2 works as follows. First, line 1 tests whether the strongly connected graph G is 2-vertex-connected using the algorithm of Italiano et

al. [ILS12], and if it is, line 2 outputs V. Otherwise, the algorithm finds a dominator tree whose depth is at least 2 as follows. Line 5 chooses a vertex $v \in V$ which is not a SAP of G. Then, line 6 computes the dominator trees $DT(v)$ and $DT^R(v)$ since at least one of them has non-trivial dominators. In order to reduce the recursion depth, we choose a dominator tree of $\{DT(v), DT^R(v)\}$ that contains more non-trivial dominators. $M(w)$ is the set of direct successors of vertex w in the dominator tree that is chosen in line 7. For each vertex $w \in V$ with $|M(w)| \geq 2$, the algorithm tests whether if $G[M(w) \cup \{w\}]$ is strongly connected, and if it is, line 17 recursively computes the 2-vccs of $G[M(w) \cup \{w\}]$. Otherwise, the for loop of lines 13–15 recursively computes the 2-vccs of $G[C]$ for every SCC C of $G[M(w) \cup \{w\}]$.

Theorem 2.3.3
Algorithm 2.3.2 runs in $O(nm)$ time.

Proof. Line 6 requires linear time [Als+99; Buc+08]. Let v, w be distinct vertices in G. Then the edge sets of the subgraphs $G[M(v) \cup \{v\}], G[M(w) \cup \{w\}]$ are disjoint since these subgraphs have at most one vertex in common. The rest of the proof is similar to the proof of Theorem 2.2.5. □

2.4. Applications

In this section, we consider three applications of our new algorithm, which are approximation algorithms for problems that are generalizations of the minimum-size 2-vertex-connected spanning subgraph problem.

Problem 2.4.1
Input: A directed graph $G = (V, E)$.
Task: Find a minimum cardinality set $E^* \subseteq E$ such that the 2-vccs of G coincide with the 2-vccs of the graph $G^* = (V, E^*)$.

Clearly, the problem of finding a minimum-size 2-vertex-connected spanning subgraph of a 2-vertex-connected directed graph is a special case of Problem 2.4.1 when G is 2-vertex-connected. Therefore, by the results from [GJ79; Geo11] Problem 2.4.1 is NP-hard.
In Figure 2.2(a), $C_1^{2vc} = \{0, 1, 2\}, C_2^{2vc} = \{2, 3, 4, 5\}, C_3^{2vc} = \{4, 6, 7\}$ and $C_4^{2vc} = \{7, 8, 9, 10, 11\}$ are the 2-vccs of the subgraph $G^d = (V, E^d)$, where $E^d =$

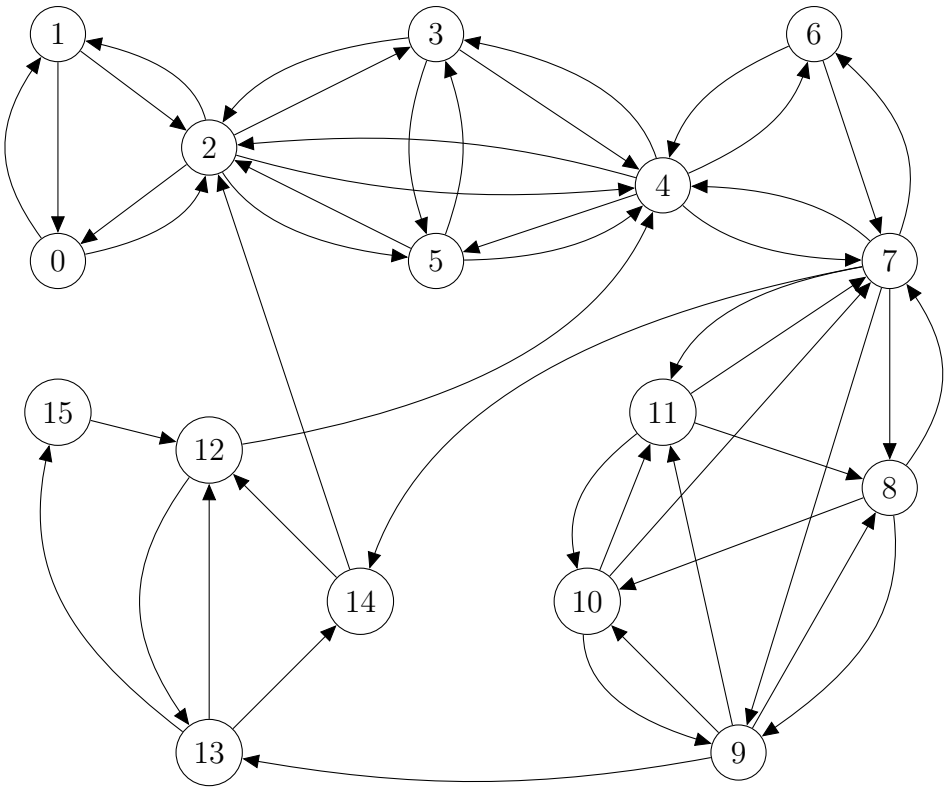
{(1,0),(0,1),(1,2),(2,1),(2,0),(0,2),(2,3),(3,2),(3,4),(4,3),(4,5),(5,4),(5,2),(2,5),(4,6),(6,4),(6,7),(7,6),(7,4),(4,7),(7,8),(8,7),(8,9),(9,8),(9,10),(10,9),(10,11),(11,10),(11,7),(7,11)}. Note that in Figure 2.2(a) the edge set E^d is an optimal solution since G^d has the same 2-vccs as G and each vertex of C_j^{2vc} has exactly indegree and outdegree 2 in the subgraph $G^d[C_j^{2vc}]$ for $1 \leq j \leq 4$. For information on lower bounds for the number of edges in a solution for the minimum-size 2-vertex-connected spanning subgraph problem, see [CT00; Geo11].

Lemma 2.4.2
There is a 1.5 approximation algorithm for Problem 2.4.1 with running time $O(nm)$.

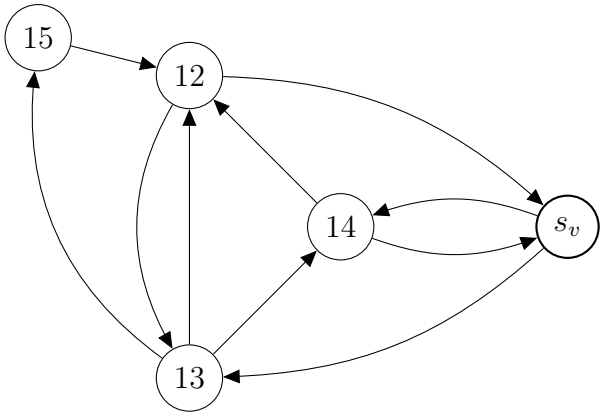
Proof. First, we compute all the 2-vccs $C_1^{2vc}, C_2^{2vc}, \ldots, C_t^{2vc}$ of the directed graph G using Algorithm 2.3.2. The edges of the set $E \setminus (E[C_1^{2vc}] \cup E[C_2^{2vc}] \cup \ldots \cup E[C_t^{2vc}])$ are irrelevant. Let E_{opt} an optimal solution for Problem 2.4.1. Then, by [ES80, Theorem 3], we have $E_{opt} = E_1 \cup E_2 \cup \ldots \cup E_t$, where E_i is an optimal solution for the subgraph $G[C_i^{2vc}]$. Let $opt = |E_{opt}|$ and $opt_i = |E_i|$. Then, $opt = \sum_{1 \leq i \leq t} opt_i$. In 2000, Cheriyan and Thurimella [CT00] gave a $(1 + 1/k)$-approximation algorithm for the problem of finding a minimum-size k-vertex-connected spanning subgraph of a directed graph with m edges. This algorithm runs in $O(km^2)$ time. In 2011, Georgiadis [Geo11] improved the running time of the algorithm of Cheriyan and Thurimella from $O(m^2)$ to $O(m\sqrt{n} + n^2)$ for $k = 2$. This improved algorithm [Geo11] preserves the 1.5 approximation guarantee of the Cheriyan-Thurimella algorithm for $k = 2$. Let $E_i^{'}$ be an edge set obtained by running the improved algorithm [Geo11] on the subgraph $G[C_i^{2vc}]$. Then, we have $\sum_{1 \leq i \leq t} |E_i^{'}| \leq 1.5 \sum_{1 \leq i \leq t} opt_i \leq 1.5 opt$ because the edge sets of $G[C_i^{2vc}], 1 \leq i \leq t$, are disjoint. The total running time is $O(\sum_{1 \leq i \leq t}(|E[C_i^{2vc}]|\sqrt{|C_i^{2vc}|} + |C_i^{2vc}|^2) + nm) = O(nm)$ because $\sum_{1 \leq i \leq t}(|E[C_i^{2vc}]|\sqrt{|C_i^{2vc}|} + |C_i^{2vc}|^2) \leq \sqrt{n} \sum_{1 \leq i \leq t} |E[C_i^{2vc}]| + \sum_{1 \leq i \leq t} |C_i^{2vc}|^2 \leq m\sqrt{n} + \sum_{1 \leq i \leq t} |C_i^{2vc}|^2$ and $O(\sum_{1 \leq i \leq t} |C_i^{2vc}|^2) = O(n^2)$ by Lemma 2.4.3. □

Lemma 2.4.3
Let $G = (V, E)$ be a directed graph and let $C_1^{2vc}, C_2^{2vc}, \ldots, C_t^{2vc}$ be the 2-vccs of G. Then $\sum_{1 \leq i \leq t} |C_i^{2vc}| < 3n$.

Proof. We construct a new undirected graph $G_c = (V_c, E_c)$ from G called a *component graph* as follows. It has a vertex v_i for each 2-vcc C_i^{2vc} of G and all



(a)



(b)

Figure 2.2.: (a) A strongly connected graph $G = (V, E)$ with its SAPs $2, 4, 7, 12$ and 13. The 2-vccs of G are $C_1^{2vc} = \{0, 1, 2\}, C_2^{2vc} = \{2, 3, 4, 5\}, C_3^{2vc} = \{4, 6, 7\}$ and $C_4^{2vc} = \{7, 8, 9, 10, 11\}$. (b) The coarsened graph of G.

vertices w that are in the intersection of at least two 2-vccs of G. Let C_i^{2vc}, C_j^{2vc} be distinct 2-vccs of G such that C_i^{2vc}, C_j^{2vc} have a vertex w in common. Then we add two undirected edges $(v_i, w), (w, v_j)$ to E_c. Since G_c is a forest, we have $\sum_{1 \leq i \leq t} |C_i^{2vc}| \leq |V| + |E_c| \leq n + n + t - 1 < 3n$. □

Problem 2.4.4
Input: A strongly connected graph $G = (V, E)$.
Task: Find a minimum cardinality set $E^* \subseteq E$ such that the 2-vccs of G coincide with the 2-vccs of the directed graph $G^* = (V, E^*)$ and G^* is strongly connected.

If we contract the 2-vccs of G that overlap into a super vertex, then we obtain a directed graph, which we call the *coarsened graph* of G, as illustrated in Figure 2.2. Note that the edge set $E^d \cup \{(9, 13), (13, 15), (15, 12), (12, 4), (13, 14), (14, 12)\}$ is an optimal solution for Problem 2.4.4 on the strongly connected graph G in Figure 2.2(a) and the edge set $\{(s_v, 13), (13, 15), (15, 12), (12, s_v), (13, 14), (14, 12)\}$ forms a minimum strongly connected spanning subgraph of the coarsened graph of G in Figure 2.2(b).

Lemma 2.4.5
There is an 5/3 approximation algorithm for Problem 2.4.4 with running time $O(nm)$.

Proof. The edge sets within the 2-vccs of G and the edge set between 2-vccs of G are disjoint. We split Problem 2.4.4 into two independent problems: Problem 2.4.1 and the minimum strongly-connected spanning subgraph problem. In 2003, Zhao et al. [ZNI03] gave a linear time 5/3-approximation algorithm for the minimum strongly-connected spanning subgraph problem. We run this algorithm on the coarsened graph of G. □

Problem 2.4.6
Input: A directed graph $G = (V, E)$.
Task: Find a minimum cardinality set $E^* \subseteq E$ such that the 2-vccs of G coincide with the 2-vccs of $G^* = (V, E^*)$ and the 2-vccs of the coarsened graph of G coincide with the 2-vccs of the coarsened graph of G^*.

Note that it can happen that the coarsened graph of a directed graph contains 2-vccs, like the 2-vccs $\{s_v, 12, 13, 14\}$ in Figure 2.2(b). Observe that any optimal solution for Problem 2.4.6 on a directed graph G consists of an optimal solution of Problem 2.4.1 on G and on the coarsened graph of G, for example $E^d \cup \{(9, 13), (7, 14), (12, 4), (14, 2), (13, 14), (14, 12), (12, 13), (13, 12)\}$ in Figure 2.2.

Lemma 2.4.7
There is an 1.5 approximation algorithm for Problem 2.4.6 with running time $O(nm)$.

Proof. The proof is similar to the proof of Lemma 2.4.2 (we apply the same method on the graph G and on the coarsened graph of G). □

2.5. An open problem

Let $G = (V, E)$ be a strongly connected graph and let v be an arbitrary vertex in G. By $K(v)$ we denote the set of direct successors of the root v in the dominator tree $DT(v)$. A vertex $w \in V$ belongs to the set $K(v)$ if and only if $(v, w) \in E$ or there exist two vertex-disjoint paths form v to w in G. We denote by $K^R(v)$ the set of direct successors of the root v in $DT^R(v)$. The following lemma shows a relationship between the 2-vccs that contain v and the dominator trees $DT(v), DT^R(v)$.

Lemma 2.5.1
Let $G = (V, E)$ be a strongly connected graph and let v be an arbitrary vertex in G. Then only elements of $((K(v) \cap K^R(v)) \cup \{v\})$ can belong to the 2-vccs which contain the vertex v.

Proof. Let $w \in V$, and assume that $w \notin (K(v) \cap K^R(v)) \cup \{v\}$, i.e. $w \notin K(v) \cap K^R(v)$ and $w \neq v$. Then $w \in (V \setminus (K(v) \cup \{v\}) \cup (V \setminus (K^R(v) \cup \{v\})$. There are two cases to consider:

1. $w \notin K(v) \cup \{v\}$. Then there exists a non-trivial dominator s such that every path from v to w includes s in G. Therefore, there are no two vertex-disjoint paths from v to w in G.

2. $w \notin K^R(v) \cup \{v\}$. We argue as in case 1. □

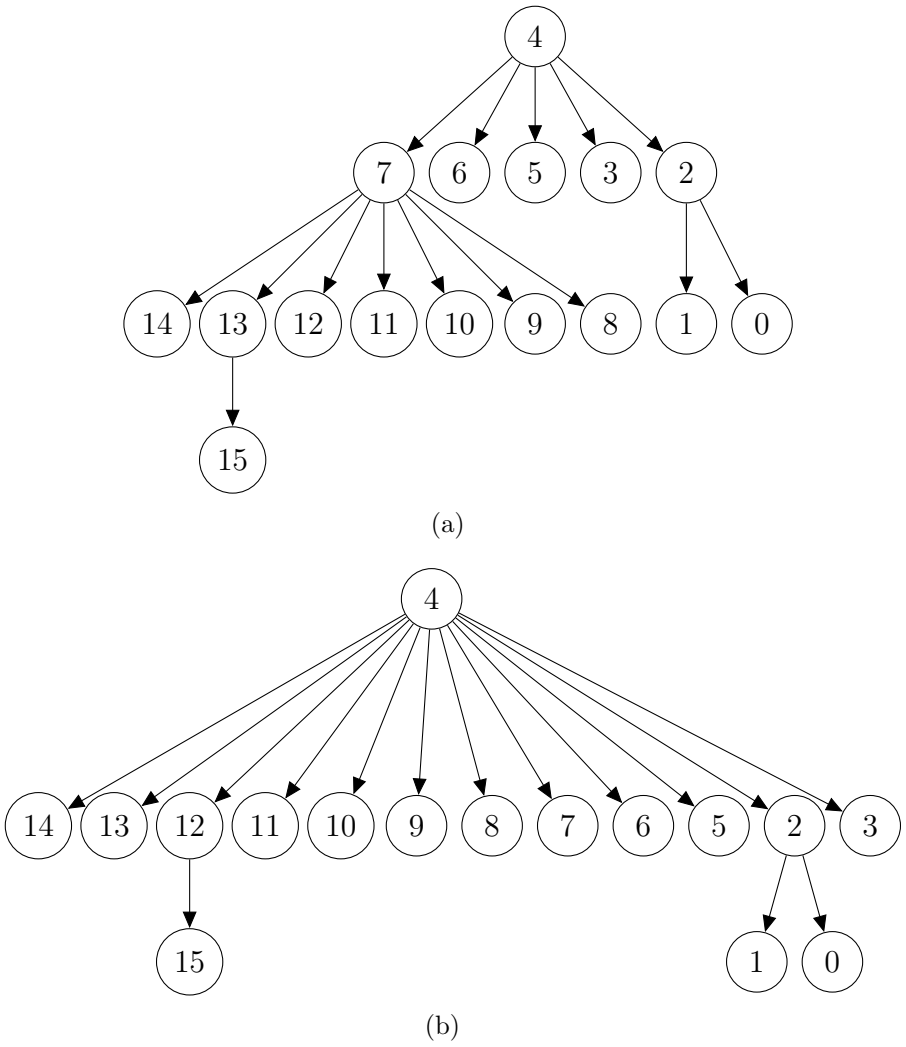


Figure 2.3.: (a) The dominator tree $DT(4)$ of the flowgraph $G(4)$ of Figure 2.2. (b) The dominator tree $DT^R(4)$ of $G^R(4)$. Note that the 2-vccs of the graph G of Figure 2.2 that contain the vertex 4 lie in the set $(K(4) \cap K^R(4)) \cup \{4\} = \{2, 3, 5, 6, 7, 4\}$.

Figure 2.3 illustrates this lemma.

We leave as an open problem whether the 2-vccs which contain a certain vertex can be computed in linear time.

3. Computing the 2-blocks of directed graphs

3.1. Computing 2-directed blocks

In this section we present our first algorithm for computing the 2-directed blocks of directed graphs. Our second algorithm will be described in section 3.4. We consider only strongly connected graphs since the 2-directed blocks of a directed graph are the union of the 2-directed blocks of its SCCs. Let $G = (V, E)$ be a strongly connected graph. For distinct vertices $x, y \in V$, we write $x \overset{2}{\rightsquigarrow} y$ if there exist two vertex-disjoint paths from x to y in G, and we write $x \overset{2}{\leftrightsquigarrow} y$ if $x \overset{2}{\rightsquigarrow} y$ and $y \overset{2}{\rightsquigarrow} x$. A 2-directed block in G is a maximal vertex set $C^{2d} \subseteq V$ with $|C^{2d}| \geq 2$ such that for each pair of distinct vertices $x, y \in C^{2d}$, we have $x \overset{2}{\leftrightsquigarrow} y$.

Lemma 3.1.1
Let $G = (V, E)$ be a strongly connected graph and let x, y be distinct vertices in G. Then $x \overset{2}{\leftrightsquigarrow} y$ if and only if for each vertex $w \in V \setminus \{x, y\}$ the vertices x, y lie in the same SCC of $G \setminus \{w\}$ and in the same SCC of $G \setminus \{(x, y), (y, x)\}$.

Proof. "$\Leftarrow$": Without loss of generality, it is sufficient to show that there are two vertex-disjoint paths from x to y in G. We consider two cases.

1. $(x, y) \notin E$. Let $w \in V \setminus \{x, y\}$. Since the vertices x, y lie in the same SCC of $G \setminus \{w\}$, there exists a path from x to y in $G \setminus \{w\}$. Thus, one can not interrupt all paths from x to y by removing w from G. Since x and y are not adjacent, by Menger's Theorem for vertex connectivity [BR12] we have $x \overset{2}{\rightsquigarrow} y$.

2. $(x, y) \in E$. Since x, y lie in the same SCC of $G \setminus \{(x, y), (y, x)\}$, there is

a path p_1 from x to y in $G \setminus \{(x, y)\}$. Thus, there are two vertex-disjoint paths p_1 and $p_2 = (x, y)$ from x to y in G.

"⇒": We know there are two vertex-disjoint paths p_1 and p_2 from x to y in G. We must show that in $G \setminus \{w\}$ and in $G \setminus \{(x, y)\}$ there is a path from x to y. Since at most one of p_1 and p_2 contains w and at most one of p_1 and p_2 is edge (x, y), the claim follows. □

Lemma 3.1.2
Let $G = (V, E)$ be a strongly connected graph and let x, y be distinct vertices in G such that $x \overset{2}{\rightsquigarrow} y$. Then the vertices x, y lie in the same SCC of $G \setminus \{e\}$ for any edge $e \in E$.

Proof. There exist two vertex-disjoint paths p_1, p_2 from x to y and two vertex-disjoint paths p_3, p_4 from y to x in G since $x \overset{2}{\rightsquigarrow} y$. The paths p_1, p_2 are edge-disjoint and the paths p_3, p_4 are also edge-disjoint. Hence, there exists a path from x to y and a path from y to x in $G \setminus \{e\}$ for any edge $e \in E$. □

Lemma 3.1.3 shows that 2-directed blocks intersect in at most one vertex. (2-vertex-connected components have the same property, see [ES80] and [Jab15b].)

Lemma 3.1.3
Let C_1^{2d}, C_2^{2d} be distinct 2-directed blocks in a strongly connected graph $G = (V, E)$. Then C_1^{2d} and C_2^{2d} have at most one vertex in common.

Proof. Suppose for a contradiction that $|C_1^{2d} \cap C_2^{2d}| > 1$. By renaming we can assume that there are at least two vertices $v \in C_1^{2d}, w \in C_2^{2d}$ with $v, w \notin C_1^{2d} \cap C_2^{2d}$ such that there are no two vertex-disjoint paths from v to w in G. We consider two cases.

1. $(v, w) \notin E$. By Menger's Theorem [BR12] there is some vertex $s \in V \setminus \{v, w\}$ such that s lies on all paths from v to w. Let z be a vertex in $(C_1^{2d} \cap C_2^{2d}) \setminus \{s\}$. Since C_1^{2d} and C_2^{2d} are 2-directed blocks, there is a path from v to z in $G \setminus \{s\}$ and a path from z to w in $G \setminus \{s\}$, hence there is a path from v to w in $G \setminus \{s\}$, which is a contradiction.

2. $(v, w) \in E$. In this case there is no path from v to w in $G \setminus \{(v, w)\}$. Let u be a vertex in $C_1^{2d} \cap C_2^{2d}$. But, again by the definition of 2-directed blocks, there are paths from v to u and from u to w in $G \setminus \{(v, w)\}$, a contradiction.

□

Next we note that 2-directed blocks can not form cycles in the following sense.

Lemma 3.1.4
Let $G = (V, E)$ be a strongly connected graph and let $v_0, v_1, \ldots, v_l$ be distinct vertices of G such that $v_0 \overset{2}{\rightsquigarrow} v_l$ and $v_{i-1} \overset{2}{\rightsquigarrow} v_i$ for $i \in \{1, 2 \ldots, l\}$. Then all the vertices $v_0, v_1, \ldots, v_l$ lie in the same 2-directed block of G.

Proof. Suppose for a contradiction that there exist two vertices v_r, v_q with $r, q \in \{0, 1, \ldots, l\}$ such that v_r, v_q lie in distinct 2-directed blocks of G and $r < q$. By renaming, we may assume that there do not exist two vertex-disjoint paths from v_r to v_q in G. We consider two cases.

1. $(v_r, v_q) \notin E$. In this case, all the paths from v_r to v_q contain a vertex $s \in V \setminus \{v_r, v_q\}$. Therefore, there is no path from v_r to v_q in $G \setminus \{s\}$. There are two cases to consider.
   a) $s \notin \{v_{r+1}, v_{r+2}, \ldots, v_{q-1}\}$. In this case, for each $i \in \{r+1, r+2, \ldots, q\}$, there is a path from v_{i-1} to v_i in $G \setminus \{s\}$ by Lemma 3.1.1, a contradiction.
   b) $s \in \{v_{r+1}, v_{r+2}, \ldots, v_{q-1}\}$. Then by Lemma 3.1.1, there are paths from v_r to $v_{r-1}, \ldots$ from v_1 to v_0, from v_0 to v_l, from v_l to $v_{l-1}, \ldots$ from v_{q+1} to v_q in $G \setminus \{s\}$, again a contradiction.
2. $(v_r, v_q) \in E$. By Lemma 3.1.2, for each $i \in \{r+1, r+2, \ldots, q\}$, the vertices v_{i-1}, v_i lie in the same SCC of $G \setminus \{(v_r, v_q)\}$. Therefore, there exists a path p_1 from v_r to v_q in $G \setminus \{(v_r, v_q)\}$. Consequently, there are two vertex-disjoint paths p_1 and $p_2 = (v_r, v_q)$ from v_r to v_q in G, but this is a contradiction.

□

We construct the 2-*directed block graph* $G^{2d} = (V^{2d}, E^{2d})$ of a strongly connected graph $G = (V, E)$ as follows. It has a vertex v_i for every 2-directed block C_i^{2d} and all vertices w that lie in the intersection of (at least) two 2-directed blocks. For each pair of distinct 2-directed blocks C_i^{2d}, C_j^{2d} with $C_i^{2d} \cap C_j^{2d} = \{w\}$, we add two undirected edges $(v_i, w), (w, v_j)$ to E^{2d}.

Lemma 3.1.5
Let $G = (V, E)$ be a strongly connected graph. Then the 2-directed block graph $G^{2d} = (V^{2d}, E^{2d})$ of G is a forest.

Proof. This follows from Lemma 3.1.4. □

Now we turn to algorithm for finding the 2-directed blocks. Algorithm 3.1.6 describes our first algorithm for computing all the 2-directed blocks of a strongly connected graph G.

Algorithm 3.1.6

Input: A strongly connected graph $G = (V, E)$.
Output: The 2-directed blocks of G.

```
1   if G is 2-vertex-connected then
2     Output V.
3   else
4     Let A be an n × n matrix.
5     Initialize A with 0s.
6     for each ordered pair (v, w) ∈ V × V do
7       if there are two vertex-disjoint paths from v to w in G then
8         A[v, w] ← 1.
9     Construct undirected graph G* = (V, E*) as follows.
10    for each pair (v, w) ∈ V × V do
11      if A[v, w] = 1 and A[w, v] = 1 then
12        Add the undirected edge (v, w) to E*.
13    Compute the blocks of size > 1 of G* = (V, E*) and output them.
```

Lemma 3.1.7
Algorithm 3.1.6 calculates 2-directed blocks.

Proof. If G is 2-vertex connected, then V is a 2-directed block. Let $G = (V, E)$ be a strongly connected graph which is not 2-vertex connected. For any vertices $v, w \in V$, $v \overset{2}{\rightsquigarrow} w$ if and only if $A[v, w] = 1$ and $A[w, v] = 1$ in line 11. Hence, $v \overset{2}{\rightsquigarrow} w$ if and only if $(v, w) \in E^*$. Let x, y be two vertices that do not lie in the same block of G^*. Then (x, y) can not be in E^*. Hence, the vertices x, y

do not lie in the same 2-directed block of G. Let B be a block of G^* containing v, w with $(v, w) \in E^*$. There are two cases to consider.

1. $B = \{v, w\}$. Then $v \overset{2}{\rightsquigarrow} w$ and $\{v, w\}$ is a 2-directed block. (If there were some z such that v, w, z are in the same 2-directed block, we would have the triangle $(v, z), (z, w), (w, v)$ in G^*, hence z would be in the same block as v, w.)

2. B contains other vertices. We show that all these vertices are in the same 2-directed block. If $z \in V \setminus \{v, w\}$ is in B, then z, v lie on one simple cycle in G^*. By Lemma 3.1.4, the vertices z, v lie in the same 2-directed block.

□

It remains to describe Procedure 3.1.8 that implements steps 6–8 of Algorithm 3.1.6.

Procedure 3.1.8

Purpose: Check if there are two vertex disjoint paths.
Input: A strongly connected graph $G = (V, E)$.
Output: Matrix A.

```
1    for each vertex v ∈ V do
2      E' ← E.
3      V' ← V.
4      for each edge e = (v, w) ∈ E do
5        E' ← E' \ {(v, w)}.
6        V' ← V' ∪ {u_e}.
7        E' ← E' ∪ {(v, u_e), (u_e, w)}.
8      Compute the dominator tree DT'(v) of the flowgraph G'(v) = (V', E', v).
9      for each direct successor w of v in DT'(v) do
10       if w ∈ V then
11         A[v, w] ← 1.
```

For each vertex $v \in V$, we construct a directed graph $G' = (V', E')$ from G as follows. For each edge $(v, w) \in E$, we remove this edge (v, w) and we add a new vertex u_e and two new edges $(v, u_e), (u_e, w)$ to G'. Then we compute the dominator tree $DT'(v)$ of the flowgraph $G'(v) = (V', E', v)$. For each direct successor w of v in $DT'(v)$ such that $w \in V$, line 11 sets $A[v, w]$ to 1. The correctness of Procedure 3.1.8 follows from the following lemma.

Lemma 3.1.9
Let $G = (V, E)$ be a strongly connected graph and let v, w be two distinct vertices in G. Then $v \overset{2}{\rightsquigarrow} w$ in $G(v)$ if and only if v is the immediate dominator of w in the flowgraph $G'(v) = (V', E', v)$.

Proof. "⇒" Assume that $v \overset{2}{\rightsquigarrow} w$ in $G(v)$. Then there are two vertex-disjoint paths $p_1 = (v = v_1, v_2, \ldots, v_t = w)$ and $p_2 = (v = u_1, u_2, \ldots, u_l = w)$ from v to w in $G(v)$. In lines 4–7 of Procedure 3.1.8, the edge $x = (v_1, v_2)$ is replaced by two edges $(v_1, v_x), (v_x, v_2)$ and the edge $y = (u_1, u_2)$ is replaced by two edges $(u_1, u_y), (u_y, u_2)$. Since $v_x \neq u_y$, there exist two vertex-disjoint paths from v to w in $G'(v)$. Therefore, v is the immediate dominator of w in the flowgraph $G'(v)$.
"⇐" We prove the contrapositive. Assume that $v \overset{2}{\rightsquigarrow} w$ in $G(v)$ is not true. Then there is some vertex $x \in V \setminus \{v, w\}$ such that all paths from v to w in $G(v)$ contain x. Then x is a non-trivial dominator of w in $G(v)$. Thus, v is not the immediate dominator of w in $G(v)$. Let $p = (v = v_1, v_2, \ldots, v_t = w)$ be a simple path from v to w in $G(v)$. In lines 4–7 of Procedure 3.1.8, $e = (v_1, v_2)$ is replaced by $(v_1, u_e), (u_e, v_2)$. Hence, the path p corresponds to the simple path $(v = v_1, u_e, v_2, \ldots, v_t = w)$ in $G'(v)$. Since $u_e \neq x$, the vertex x is a non-trivial dominator of w in $G'(v)$. Therefore, v is not the immediate dominator of w in $G'(v)$. □

Remark 3.1.10
Procedure 3.1.8 checks in polynomial time whether there are two vertex-disjoint paths from v to w in G. It may be worth noticing that problems of a similar flavor are NP-complete: Fortune et al. [FHW80] proved it is NP-complete to check if there are vertex-disjoint (arc-disjoint) paths from s_1 to t_1 and from s_2 to t_2 for four given vertices. Li et al. [LMS90] showed it is NP-hard to find two vertex-disjoint (arc-disjoint) paths from s to t while minimizing the length of the longer one.

Theorem 3.1.11
Algorithm 3.1.6 runs in $O(nm)$ time.

Proof. The dominators of a flowgraph can be found in linear time [Buc+08; Als+99]. In lines 2–7 of Procedure 3.1.8, the construction of $G'(v) = (V', E', v)$

takes linear time because the graph G' has $|V'| = n + d_{out}(v) < 2n$ vertices and $|E'| = m + d_{out}(v) < m + n$ edges. Moreover, lines 9–11 of Procedure 3.1.8 take $O(n)$ time since the number of direct successors of v in the dominator tree $DT'(v)$ is at most $2(n-1)$. Since the number of iterations of the for loop in lines 1–11 of Procedure 3.1.8 is n, the running time of Procedure 3.1.8 is $O(nm)$. One can test whether a directed graph is 2-vertex-connected in linear time using the algorithm of Italiano et al. [ILS12]. The initialization of matrix A requires $O(n^2)$ time. The undirected graph $G^* = (V, E^*)$ can also be constructed in $O(n^2)$ time. Furthermore, the blocks of an undirected graph can be computed in linear time using Tarjan's algorithm [Tar72]. The total cost is therefore $O(nm + n^2) = O(nm)$. □

Let $G = (V, E)$ be a strongly connected graph. By definition, the 2-directed blocks of G are the maximal cliques of the auxiliary graph G^* which is constructed in lines 4–12 of Algorithm 3.1.6. By Lemma 3.1.4, the auxiliary graph G^* is chordal. In line 13 of Algorithm 3.1.6, one can compute the maximal cliques of the auxiliary graph G^* instead of blocks since the maximal cliques of a chordal graph can be calculated in linear time [Gav72; RT75].

3.2. Computing 2-strong blocks

In this section we present two algorithms for computing the 2-strong blocks of directed graphs. The 2-strong blocks of a directed graph are the union of the 2-strong blocks of its SCCs. Let $G = (V, E)$ be a strongly connected graph. We define a relation $\overset{2s}{\leftrightsquigarrow}$ as follows. For any distinct vertices $x, y \in V$, we write $x \overset{2s}{\leftrightsquigarrow} y$ if for any vertex $z \in V \setminus \{x, y\}$, the vertices x, y lie in the same SCC of $G \setminus \{z\}$. A 2-strong block in G is a maximal vertex set $C^{2s} \subseteq V$ with $|C^{2s}| \geq 2$ such that for each pair of distinct vertices $x, y \in C^{2s}$, we have $x \overset{2s}{\leftrightsquigarrow} y$.

Let v, w be distinct vertices in V such that $(v, w) \in E$ and $w \overset{2}{\rightsquigarrow} v$. While v, w are in one 2-strong block, these vertices do not necessarily lie in the same 2-directed block of G.

Lemma 3.2.1
Each 2-directed block in a strongly connected graph G is a subset of a 2-strong block in G.

Proof. Immediate from Lemma 3.1.1. □

Lemma 3.2.2
Let $G = (V, E)$ be a strongly connected graph. Let C_1^{2s}, C_2^{2s} be distinct 2-strong blocks in G. Then C_1^{2s} and C_2^{2s} have at most one vertex in common.

Proof. Assume for a contradiction that $|C_1^{2s} \cap C_2^{2s}| > 1$. Then there exist at least two vertices $x \in C_1^{2s}, y \in C_2^{2s}$ with $x, y \notin C_1^{2s} \cap C_2^{2s}$ and a vertex $z \in V \setminus \{x, y\}$ such that the vertices x, y lie in different SCCs of $G \setminus \{z\}$. Let w be a vertex in $(C_1^{2s} \cap C_2^{2s}) \setminus \{z\}$. Since $x, w \in C_1^{2s}$, these vertices lie in the same SCC of $G \setminus \{z\}$, similarly for $w, y \in C_2^{2s}$. Hence x, y lie in the same SCC of $G \setminus \{z\}$, a contradiction. □

As with 2-directed blocks, there can not be cycles of 2-strong blocks.

Lemma 3.2.3
Let $G = (V, E)$ be a strongly connected graph and let $v_0, v_1, \ldots, v_l$ be distinct vertices of G such that $v_0 \overset{2s}{\leftrightsquigarrow} v_l$ and $v_{i-1} \overset{2s}{\leftrightsquigarrow} v_i$ for $i \in \{1, 2 \ldots, l\}$. Then all the vertices $v_0, v_1, \ldots, v_l$ lie in the same 2-strong block of G.

Proof. Let v_r, v_q be two vertices such that $r, q \in \{0, 1, \ldots, l\}$ and $r < q$. Let w be a vertex in $V \setminus \{v_r, v_q\}$. We consider two cases.

1. $w \notin \{v_{r+1}, v_{r+2}, \ldots, v_{q-1}\}$. Then, for each $i \in \{r+1, r+2, \ldots, q\}$, the vertices v_{i-1}, v_i lie in the same SCC of $G \setminus \{w\}$. Thus the vertices v_r, v_q lie in the same SCC of $G \setminus \{w\}$.

2. $w \in \{v_{r+1}, v_{r+2}, \ldots, v_{q-1}\}$. Then the vertices v_{i-1}, v_i lie in the same SCC of $G \setminus \{w\}$ for each $i \in \{1, 2, \ldots, r\} \cup \{q+1, q+2, \ldots, l\}$. Furthermore, the vertices v_0, v_l lie in the same SCC of $G \setminus \{w\}$ since $v_0 \overset{2s}{\leftrightsquigarrow} v_l$. Thus the vertices v_r, v_q lie in the same SCC of $G \setminus \{w\}$.

Since the vertices v_r, v_q lie in the same SCC of $G \setminus \{w\}$ for any vertex $w \in V \setminus \{v_r, v_q\}$, the vertices v_r, v_q lie in the same 2-strong block of G. □

Algorithm 3.2.4 shows our first algorithm for computing the 2-strong blocks of a strongly connected graph $G = (V, E)$.

Using arguments similar to those in the proof of Lemma 3.1.7, one can show that Algorithm 3.2.4 is correct.

Algorithm 3.2.4

Input: A strongly connected graph $G = (V, E)$.
Output: The 2-strong blocks of G.

```
1   if G is 2-vertex-connected then
2     Output V.
3   else
4     Let A be an n × n matrix.
5     Initialize A with 0s.
6     for each vertex v ∈ V do
7       Compute DT(v).
8       for each direct successor w of v in DT(v) do
9         A[v, w] ← 1.
10    Construct undirected graph G* = (V, E*) as follows.
11    for each pair (v, w) ∈ V × V do
12      if A[v, w] = 1 and A[w, v] = 1 then
13        Add the undirected edge (v, w) to E*.
14    Compute the blocks of size > 1 of G* = (V, E*) and output them.
```

Theorem 3.2.5
Algorithm 3.2.4 runs in $O(nm)$ time.

Proof. The dominators of a flowgraph can be found in linear time [Buc+08; Als+99]. Therefore, lines 6–9 take $O(nm)$ time. □

Lemma 3.2.6
Let $G = (V, E)$ be a strongly connected graph and let x, y be distinct vertices in G. Let S be the set of all the SAPs in G. Then for any vertex $z \in V \setminus (S \cup \{x, y\})$, the vertices x and y lie in the same SCC of $G \setminus \{z\}$.

Proof. Immediate from the definition. □

This simple lemma gives rise to an alternative algorithm (Algorithm 3.2.7) that might be helpful if the number of the SAPs is small.

Algorithm 3.2.7

Input: A strongly connected graph $G = (V, E)$.
Output: The 2-strong blocks of G.

```
1   if G is 2-vertex-connected then
2     Output V.
3   else
4     Let A be an n × n matrix.
5     Initialize A with 1s.
6     Compute the SAPs of G.
7     for each s ∈ SAPs of G do
8       Compute the SCCs of G \ {s}.
9       for each pair (v, w) ∈ (V \ {s}) × (V \ {s}) do
10        if v, w in different SCCs of G \ {s} then
11          A[v, w] ← 0.
12    E* ← ∅.
13    for each pair (v, w) ∈ V × V do
14      if A[v, w] = 1 and A[w, v] = 1 then
15        Add the undirected edge (v, w) to E*.
16    Compute the blocks of size > 1 of G* = (V, E*) and output them.
```

Lemma 3.2.8

Let v, w be distinct vertices in a strongly connected graph G. Then $v \overset{2s}{\leftrightsquigarrow} w$ if and only if $A[v, w] = 1$ and $A[w, v] = 1$ (when line 14 is reached).

Proof. "$\Leftarrow$" If $A[v, w] = 1$ and $A[w, v] = 1$, then the vertices v, w lie in the same SCC of $G \setminus \{s\}$ for any SAP $s \in V \setminus \{v, w\}$ (see lines 7–11). By Lemma 3.2.6, the vertices v, w lie in the same SCC of $G \setminus \{z\}$ for any vertex $z \in V \setminus \{v, w\}$. "$\Rightarrow$" This follows from Lemma 3.2.6. □

Theorem 3.2.9

The running time of Algorithm 3.2.7 is $O(t_{sap} n^2)$.

Proof. The SAPs of a directed graph can be computed in linear time using the algorithm of Italiano et al. [ILS12]. Lines 7–11 take $O(t_{sap} n^2)$ time. □

Corollary 3.2.10
The 2-strong blocks of a directed graph $G = (V, E)$ can be computed in $O(\min\{m, t_{sap}n\}n)$ time.

3.3. Computing 2-edge blocks

In this section we present two algorithms for computing the 2-edge blocks of directed graphs. The 2-edge blocks of a directed graph are the union of the 2-edge blocks of its SCCs. We define a relation $\overset{2e}{\leftrightsquigarrow}$ as follows. For any distinct vertices $x, y \in V$, we write $x \overset{2e}{\leftrightsquigarrow} y$ if there exist two edge-disjoint paths from x to y and two edge-disjoint paths from y to x in G. A 2-edge block in G is a maximal vertex set $C^{2e} \subseteq V$ with $|C^{2e}| \geq 2$ such that for each pair of distinct vertices $x, y \in C^{2e}$, we have $x \overset{2e}{\leftrightsquigarrow} y$.

Lemma 3.3.1
Let $G = (V, E)$ be a strongly connected graph and let x and y be distinct vertices in G. Then $x \overset{2e}{\leftrightsquigarrow} y$ if and only if for each edge $(v, w) \in E$, the vertices x, y lie in the same SCC of $G \setminus \{(v, w)\}$.

Proof. This is an immediate consequence of Menger's Theorem for edge connectivity [BR12]. □

Lemma 3.3.2
Let $G = (V, E)$ be a strongly connected graph. The 2-edge blocks of G are disjoint.

Proof. Let C_1^{2e}, C_2^{2e} be two distinct 2-edge blocks of G. Assume for a contradiction that $C_1^{2e} \cap C_2^{2e} \neq \emptyset$. Then by Lemma 3.3.1, there are two vertices $x \in C_1^{2e}, y \in C_2^{2e}$ with $x, y \notin C_1^{2e} \cap C_2^{2e}$ and an edge $(v, w) \in E$ such that the vertices x, y lie in distinct SCCs of $G \setminus \{(v, w)\}$. Let z be a vertex in $C_1^{2e} \cap C_2^{2e}$. By Lemma 3.3.1, the vertices x, z lie in the same SCC of $G \setminus \{(v, w)\}$ since C_1^{2e} is a 2-edge block and the vertices z, y lie in the same SCC of $G \setminus \{(v, w)\}$ since C_2^{2e} is a 2-edge block. Hence x, y lie in the same SCC of $G \setminus \{(v, w)\}$, a contradiction. □

Algorithm 3.3.3 shows our first algorithm for computing the 2-edge blocks of a strongly connected graph G.

Algorithm 3.3.3

Input: A strongly connected graph $G = (V, E)$.
Output: The 2-edge blocks of G.

```
1   if G is 2-edge-connected then
2     Output V.
3   else
4     Let A be an n × n matrix.
5     Initialize A with 0s.
6     for each vertex v ∈ V do
7       Compute the edge dominators of G(v) = (V, E, v).
8       for each vertex w ∈ V \ {v} do
9         If there is no edge dominator of w then
10          A[v, w] ← 1.
11    E* ← ∅.
12    for each pair (v, w) ∈ V × V do
13      if A[v, w] = 1 and A[w, v] = 1 then
14        Add the undirected edge (v, w) to E*.
15    Compute the connected components of size > 1 of the graph
16    G* = (V, E*) and output them.
```

Algorithm 3.3.3 works as follows. First, line 1 tests whether G is 2-edge-connected, and if it is, line 2 outputs V, since every 2-edge connected directed graph is a 2-edge block. Otherwise, for each vertex v in G, the algorithm computes the edge dominators of the flowgraph $G(v) = (V, E, v)$, and for each vertex $w \in V \setminus \{v\}$, line 10 sets $A[v, w]$ to 1 if there is no edge dominator of w. Let v, w be distinct vertices in G. Then $v \overset{2e}{\nleftrightarrow} w$ if and only if $A[v, w] = 1$ and $A[w, v] = 1$ in line 13. Lines 11–14 constructs an undirected graph $G^* = (V, E^*)$ as follows. For each pair $(v, w) \in V \times V$, we add an undirected edge (v, w) to E^* if $A[v, w] = 1$ and $A[w, v] = 1$. Finally, the algorithm finds the connected components of size at least 2 of G^*. This is correct by Lemma 3.3.2.

In [ILS12], Italiano et al. presented two algorithms for calculating the strong bridges of a strongly connected graph $G = (V, E)$. We use them to implement lines 8–10 of Algorithm 3.3.3 as follows. Consider a flowgraph $G(v) = (V, E, v)$. For each edge $e = (x, y) \in E$, we delete this edge from $G(v)$ and we add two new edges $(x, u_e), (u_e, y)$ to $G(v)$. We obtain a new flowgraph, denoted

$G'(v) = (V', E', v)$. Then, we compute the dominator tree $DT'(v)$ of $G'(v)$. Obviously, an edge e is an edge dominator of vertex $w \in V \setminus \{v\}$ in $G(v)$ if and only if the corresponding vertex u_e is a dominator of w in $G'(v)$. We mark the vertices of G that have edge dominators in $G(v)$ by depth first search in $DT'(v)$. Therefore, lines 8–10 can be implemented in linear time. In [ILS12], Italiano et al. observed that the strong bridges of G are the SAPs of the directed graph $G' = (V', E')$ that correspond to edges in G. We will use these strong bridges in our second algorithm for computing the 2-edge blocks of G.

Theorem 3.3.4
Algorithm 3.3.3 runs in $O(nm)$ time.

Proof. One can test whether a directed graph is 2-edge-connected in linear time using the algorithm of Italiano et al. [ILS12]. Furthermore, the edge dominators of a flowgraph $G(v) = (V, E, v)$ can be computed in linear time [ILS12; Fir+12]. Lines 6–10 take $O(nm)$ time. The connected components of G^* can be found in $O(n^2)$ time. □

Lemma 3.3.5
Let $G = (V, E)$ be a strongly connected graph and let x, y be distinct vertices in G. Let S_{sb} be the set of all the strong bridges of G. Then for any edge $e \in E \setminus S_{sb}$, the vertices x and y lie in the same SCC of $G \setminus \{e\}$.

Proof. Immediate from the definition. □

This simple lemma leads to another algorithm (Algorithm 3.3.6) which might be useful when t_{sb} is small.

Algorithm 3.3.6

Input: A strongly connected graph $G = (V, E)$.
Output: The 2-edge blocks of G.

```
1   If G is 2-edge-connected then.
2     Output V.
3   else
4     Let A be an n × n matrix.
5     Initialize A with 1s.
6     for each strong bridge e of G do
7       for each pair (v, w) ∈ V × V do
```

```
8          if v, w in distinct SCCs of G \ {e} then
9            A[v, w] ← 0.
10     E* ← ∅.
11     for each pair (v, w) ∈ V × V do
12       if A[v, w] = 1 and A[w, v] = 1 then
13         Add the undirected edge (v, w) to E*.
14     Compute the connected components of size > 1 of G* and output
15     them.
```

The correctness of this algorithm follows from the following lemma.

Lemma 3.3.7

Let v, w be distinct vertices in a strongly connected graph G. Then $v \overset{2e}{\leftrightsquigarrow} w$ if and only if $A[v, w] = 1$ and $A[w, v] = 1$ (when line 10 is reached).

Proof. Similar to the proof of Lemma 3.2.8 using Lemma 3.3.5. □

Theorem 3.3.8

Algorithm 3.3.6 runs in $O(t_{sb}n^2)$ time.

Proof. The strong bridges of a directed graph can be found in linear time using the algorithm of Italiano et al. [ILS12]. Lines 6–9 take $O(t_{sb}n^2)$ time. □

Let G a directed graph. Italiano et al. [ILS12] showed that $t_{sb} \leq 2n - 2$.

Corollary 3.3.9

The 2-edge blocks of a directed graph $G = (V, E)$ can be computed in $O(\min\{m, t_{sb}n\}n)$ time.

Now we show that the 2-edge block that contains a certain vertex can be computed in linear time. Let $G = (V, E)$ be a strongly connected graph and let $v \in V$. By $U(v)$ we denote the set of vertices that do not have edge dominators in $G(v)$ and by $U^R(v)$ we denote the set of vertices that do not have edge dominators in $G^R(v)$.

Lemma 3.3.10
Let $G = (V, E)$ be a strongly connected graph and let $v \in V$. Let C^{2e} be the 2-edge block of G that includes v. Then $w \in C^{2e}$ if and only if $w \in U(v) \cap U^R(v)$

Proof. "⇐" Let $w \in (U(v) \cap U^R(v)) \setminus \{v\}$. w does not have any edge dominator in $G(v)$. Therefore, by Menger's Theorem for edge connectivity, there exist two edge-disjoint paths from v to w in $G(v)$. Furthermore, there are two edge-disjoint paths from v to w in $G^R(v)$ since w does not have any edge dominator in $G^R(v)$. Thus, there are also two edge-disjoint paths from w to v in G.
"⇒" Immediate from definition. □

We have seen that $U(v)$ can be computed in linear time. Therefore, $U(v) \cap U^R(v)$ can be computed in linear time.

3.4. The relation between 2-directed blocks, 2-strong blocks and 2-edge blocks

In this section we consider the relation between 2-directed blocks, 2-strong blocks and 2-edge blocks.

Lemma 3.4.1
Let $G = (V, E)$ be a strongly connected graph and let x, y be distinct vertices in G. Then $x \overset{2}{\rightsquigarrow} y$ if and only if $x \overset{2s}{\rightsquigarrow} y$ and $x \overset{2e}{\rightsquigarrow} y$.

Proof. "⇐": By Lemma 3.3.1, for each edge $e \in E$ the vertices x, y lie in the same SCC of $G \setminus \{e\}$ since $x \overset{2e}{\rightsquigarrow} y$. Because the vertices x, y lie in the same SCC of $G \setminus \{(x, y)\}$, there exist a path from x to y in $G \setminus \{(x, y)\}$. There is also a path from y to x in $G \setminus \{(y, x)\}$ since x, y lie in the same SCC of $G \setminus \{(y, x)\}$. As a consequence, the vertices x, y lie in the same SCC of $G \setminus \{(x, y), (y, x)\}$. By definition, the vertices x, y lie in the same SCC of $G \setminus \{w\}$ for any vertex $w \in V \setminus \{x, y\}$ since $x \overset{2s}{\rightsquigarrow} y$. Therefore, by Lemma 3.1.1, we have $x \overset{2}{\rightsquigarrow} y$.
"⇒": This direction follows from Lemma 3.1.2 and Lemma 3.2.1. □

Now we describe our second algorithm for computing all the 2-directed blocks of a strongly connected graph G. First, we execute lines 1–11 of Algorithm 3.2.7. Next, we execute lines 6–9 of Algorithm 3.3.6. Finally, we execute lines

12–16 of Algorithm 3.2.7. The correctness of our algorithm follows from Lemma 3.4.1.

Theorem 3.4.2
The 2-directed blocks of a directed graph G can be computed in $O((t_{sap} + t_{sb})n^2)$ time.

Proof. This follows from Theorem 3.2.9 and Theorem 3.3.8. □

Corollary 3.4.3
The 2-directed blocks of a directed graph $G = (V, E)$ can be computed in $O(\min\{m,(t_{sap} + t_{sb})n\}n)$ time.

Theorem 3.4.4
All algorithms in Sections 3.1, 3.2, 3.3 and 3.4 require $\Theta(n^2)$ space.

Proof. Clearly, all these algorithms get by with $O(n^2)$ space and they need $\Omega(n^2)$ space to store the matrix A and the auxiliary graph G^*. □

3.5. The 2-directed blocks that contain a certain vertex

Let $G = (V, E)$ be a strongly connected graph and let v be a vertex in G. In this section we present an algorithm for computing the 2-directed blocks of G that contain v in $O(t^*m)$ time, where t^* is the number of these blocks. This algorithm is based on Lemma 3.1.3 and Lemma 3.1.4. It offers two advantages, first, it does not need to construct the auxiliary graph G^*. Second, it runs in linear time when v is contained in a constant number of 2-directed blocks. By $B(v)$ we denote the set of all vertices $w \in V \setminus \{v\}$ such that $v \overset{2}{\rightsquigarrow} w$. One can compute $B(v)$ by using Procedure 3.5.1 in linear time.

Procedure 3.5.1

Input: A strongly connected graph $G = (V, E)$ and vertex $v \in V$.
Output: $B(v)$.
1 $B_1(v) \leftarrow \emptyset$, $B_2(v) \leftarrow \emptyset$, $B(v) \leftarrow \emptyset$.
2 $E' \leftarrow E$.

```
3   V' ← V.
4   for each edge e = (v, w) ∈ E do
5     E' ← E' \ {(v, w)}.
6     V' ← V' ∪ {u_e}.
7     E' ← E' ∪ {(v, u_e), (u_e, w)}.
8   Compute the dominator tree DT'(v) of the flowgraph G'(v) = (V', E', v).
9   for each direct successor w of v in DT'(v) do
10    if w ∈ V then
11      B_1(v) ← B_1(v) ∪ {w}.
12  E_1 ← E^R.
13  V_1 ← V.
14  for each edge e = (v, w) ∈ E^R do
15    E_1 ← E_1 \ {(v, w)}.
16    V_1 ← V_1 ∪ {u_e}.
17    E_1 ← E_1 ∪ {(v, u_e), (u_e, w)}.
18  Compute the dominator tree DT_1(v) of G_1(v) = (V_1, E_1, v).
19  for each direct successor w of v in DT_1(v) do
20    if w ∈ V then
21      B_2(v) ← B_2(v) ∪ {w}.
22  B(v) ← B_1(v) ∩ B_2(v).
```

The correctness of Procedure 3.5.1 follows from Lemma 3.1.9 and the fact that $w \overset{2}{\rightsquigarrow} v$ in G if and only if $v \overset{2}{\rightsquigarrow} w$ in G^R.

Algorithm 3.5.2

Input: A strongly connected graph $G = (V, E)$ and vertex $v \in V$.
Output: The 2-directed blocks of G that contain v.

```
1   if G is 2-vertex-connected then
2     Output V.
3   else
4     R ← B(v).
5     while R is not empty do
6       Choose arbitrarily a vertex w ∈ R.
7       output (R ∩ B(w)) ∪ {v, w}.
8       R ← R \ ((R ∩ B(w)) ∪ {w}).
```

Lemma 3.5.3
Algorithm 3.5.2 calculates the 2-directed blocks that include v.

Proof. Let $C_1^{2d}, C_2^{2d}, \ldots, C_t^{2d}$ be the 2-directed blocks which contain v. By Lemma 3.1.3, these blocks include only the vertex v in common. Thus, $C_1^{2d} \setminus \{v\}, C_2^{2d} \setminus \{v\}, \ldots, C_t^{2d} \setminus \{v\}$ are disjoint. Obviously, $\bigcup_{1 \leq i \leq t}(C_i^{2d}) \setminus \{v\} \subseteq B(v)$. Let w be a vertex in $B(v)$ and let C^{2d} be the 2-directed block of G such that $v, w \in C^{2d}$. It is sufficient to show that $C^{2d} = (B(w) \cap B(v)) \cup \{v, w\}$. Let x be a vertex in $B(w) \cap B(v)$. Since $v \overset{2}{\rightsquigarrow} w$, $w \overset{2}{\rightsquigarrow} x$ and $x \overset{2}{\rightsquigarrow} v$, by Lemma 3.1.4, the vertices x, v, w lie in the same 2-directed block of G. Conversely, let x be a vertex in $C^{2d} \setminus \{v, w\}$. Since $v \overset{2}{\rightsquigarrow} x$ and $w \overset{2}{\rightsquigarrow} x$, we have $x \in B(v)$ and $x \in B(w)$. □

Theorem 3.5.4
Algorithm 3.5.2 runs in $O(t^*m)$, where t^* is the number of the 2-directed blocks that contain v.

Proof. We have seen that $B(v)$ can be computed in linear time. Furthermore, the number of iterations of the while-loop in lines 5–8 is t^*. Thus, the total time is $O(t^*m)$. □

3.6. Approximation algorithm for the MS-SAPs Problem

In this section we show that there is a 17/3 approximation algorithm for the MS-SAPs problem. In [Geo11], Georgiadis presented a linear time 3-approximation algorithm for the problem of finding a minimum-cardinality 2-vertex connected spanning subgraph (2VCSS) of 2-vertex-connected directed graphs. This algorithm is based on the works [Geo10; GT05; ILS12]. We slightly modify this algorithm and combine it with the algorithm of Zhao et al. [ZNI03] in order to obtain a 17/3 approximation algorithm for the MS-SAPs problem. We first briefly describe Georgiadis algorithm [Geo11]. Let $G = (V, E)$ be a 2-vertex-connected directed graph and let v be a vertex in G. Menger's Theorem for vertex connectivity [BR12] implies that the flowgraph $G(v)$ has no non-trivial dominators. In [GT05], Georgiadis and Tarjan proved that there exist two

Algorithm 3.6.1
(from [Geo11])
Input: A 2-vertex-connected directed graph $G = (V, E)$.
Output: A 2-vertex-connected spanning subgraph G^* of G.
1 Choose arbitrarily a vertex $v \in V$.
2 Compute two independent spanning trees T_1, T_2 of $G(v)$.
3 Compute two independent spanning trees T_3, T_4 of $G^R(v)$.
4 Construct a strongly connected spanning subgraph (SCSS)
6 $G' = (V \setminus \{v\}, E')$ of $G \setminus \{v\}$ with $|E'| \leq 2(n-2)$.
7 $E^* \leftarrow T_1 \cup T_2 \cup T_3^R \cup T_4^R \cup E'$.
8 Output $G^* = (V, E^*)$.

independent spanning trees of $G(v)$. Algorithm 3.6.1 shows the algorithm of Georgiadis [Geo11].

By [Geo11, Lemma 2], the flowgraphs $(V, T_1 \cup T_2, v)$ and $(V, T_3 \cup T_4, v)$ have only trivial dominators. Let w be a vertex in $G \setminus \{v\}$. As is well known, it is easy to calculate a SCSS $G' = (V \setminus \{v\}, E')$ of $G \setminus \{v\}$ with $|E'| \leq 2(n-2)$. Just take the union of outgoing branching rooted at w and incoming branching rooted at w ([FJ81; KRY94]). Since $G^* \setminus \{v\}$ is strongly connected, the vertex v is not a SAP in G^*. Therefore, by [ILS12, Theorem 5.2] the directed graph G^* has no SAPs. By definition, G^* is 2-vertex-connected. Algorithm 3.6.1 has an approximation ratio of 3 and runs in linear time [Geo11].

In [ZNI03], Zhao et al. gave a linear time 5/3 approximation algorithm for the problem of finding a MSCSS of a strongly connected graph. We briefly describe this algorithm. The algorithm of Zhao et al. [ZNI03] is based on repeatedly contracting special cycles. The idea of contracting cycles was first introduced by Khuller et al. [KRY94]. Let $G = (V, E)$ be a strongly connected graph and let $U \subseteq V$. By $\delta^+(U)$ we denote the set of edges leaving U, i.e., $\delta^+(U) = \{(u, v) \in E | u \in U \text{ and } v \notin U\}$. By G/U we denote the directed graph obtained from G by contracting the vertex set U. Let l be a cycle of G. By V_l and E_l we denote the set of vertices and the set of edges of the cycle l, respectively. The cycle l conceals U if $\delta^+(U)$ is not empty and the edges in $\delta^+(U)$ are deleted in G/V_l. The algorithm of Zhao et al. [ZNI03] repeatedly contracts concealing cycles. Algorithm 3.6.2 shows a high-level description of this algorithm. Zhao et al. [ZNI03] proved that Algorithm 3.6.2 returns a feasible solution for the MSCSS problem and has an approximation factor of

Algorithm 3.6.2
(from [ZNI03])
Input: A strongly connected graph $G = (V, E)$.
Output: A SCSS $G^* = (V, E^*)$ of G.
1 $E^* \leftarrow \emptyset$.
2 $G_1 \leftarrow G$.
3 **while** G_1 has a cycle of length at least 3 **do**
4 Find a concealing cycle l in G_1 with $E_l \geq 3$.
5 $E^* \leftarrow E^* \cup E_l$.
6 $G_1 \leftarrow G_1/V_l$.
7 $E_1 \leftarrow$ the set of edges of G_1.
8 $E^* \leftarrow E^* \cup E_1$.
9 Output $G^* = (V, E^*)$.

5/3. In [ZNI03], they also showed that Algorithm 3.6.2 can be implemented in linear time.

Now we modify Georgiadis's algorithm [Geo11] and combine it with the algorithm of Zhao et al. [ZNI03] in order to obtain an approximation algorithm for the MS-SAPs problem. See Algorithm 3.6.3.

Algorithm 3.6.3

Input: A strongly connected graph $G = (V, E)$.
Output: A SCSS $G^* = (V, E^*)$ of G with the same SAPs.
1 **if** G is 2-vertex-connected **then**
2 Run Algorithm 3.6.1 on G.
3 **else**
4 Compute the SAPs of G.
5 **If** all vertices in V are SAPs **then**
6 Compute a SCSS $G^* = (V, E^*)$ of G using the Algorithm of
7 Zhao et al. [ZNI03] and output G^*.
8 **else**
9 Choose a vertex $v \in V$ such that v is not a SAP of G.
10 Compute a SCSS $G' = (V \setminus \{v\}, E')$ of $G \setminus \{v\}$ using the Algorithm
11 of Zhao et al. [ZNI03].
12 Compute two independent spanning trees T_1, T_2 of $G(v)$.

13 Compute two independent spanning trees T_3, T_4 of $G^R(v)$.
14 $E^* \leftarrow E' \cup T_1 \cup T_2 \cup T_3^R \cup T_4^R$.
15 Output $G^* = (V, E^*)$.

The following lemma shows that the output G^* of Algorithm 3.6.3 is a feasible solution for the MS-SAPs problem.

Lemma 3.6.4
The output G^* is strongly connected, and the directed graphs G^* and G have the same SAPs.

Proof. We need only to show that this lemma is correct when G is not 2-vertex-connected. Let G be a strongly connected graph which is not 2-vertex-connected. If all the vertices in V are SAPs in G, then, for any SCSS G' of G, the graphs G' and G have the same SAPs. Otherwise, G has at least one vertex v which is not a SAP of G (see lines 9–14). In this case, by [ILS12, Theorem 5.2], $D(v) \cup D^R(v)$ is the set of all SAPs of G. Georgiadis and Tarjan [GT05] showed that there exist two independent spanning trees of $G(v)$ and the flowgraphs $G(v)$ and $(V, T_1 \cup T_2, v)$ have the same non-trivial dominators. Thus, the flowgraphs $G^R(v)$ and $(V, T_3 \cup T_4, v)$ have the same non-trivial dominators. Clearly, $(V, T_1 \cup T_3^R)$ is strongly connected. Therefore, G^* is strongly connected. Let $D_1(v)$ be the set of all non-trivial dominators of $G^*(v)$ and let $D_1^R(v)$ be the set of all non-trivial dominators of $G^{*R}(v)$, where G^{*R} is the reversal graph of G^*. Since $T_1 \cup T_2 \subseteq E^*$, we have $D_1(v) \subseteq D(v)$. Moreover, $D_1^R(v) \subseteq D^R(v)$ because $T_3^R \cup T_4^R \subseteq E^*$. As a consequence, $D_1(v) \cup D_1^R(v) \subseteq D(v) \cup D^R(v)$. Assume for a contradiction that $D_1(v) \cup D_1^R(v) \neq D(v) \cup D^R(v)$. Then there is at least vertex $x \in D(v) \cup D^R(v)$ such that $x \notin D_1(v) \cup D_1^R(v)$. By [ILS12, Theorem 5.2], the vertex x is not a SAP in G^* but x is a SAP in G. Thus, $G^* \setminus \{x\}$ is strongly connected. Because $G^* \setminus \{x\}$ is a spanning subgraph of $G \setminus \{x\}$, the directed graph $G \setminus \{x\}$ is strongly connected, which contradicts that x is a SAP in G. Since v is not a SAP of G^* and $D(v) \cup D^R(v) = D_1(v) \cup D_1^R(v)$, by [ILS12, Theorem 5.2] the set of all SAPs of G^* is $D(v) \cup D^R(v)$. □

Theorem 3.6.5
Algorithm 3.6.3 achieves an approximation ratio of 17/3.

Proof. The algorithm of Zhao et al. [ZNI03] has an approximation factor of 5/3. Thus, it is enough to consider the case when G is not 2-vertex-connected and

includes at least one vertex which is not a SAP. Let E_{opt} be an optimal solution for the MS-SAPs problem. Let E'_{opt} be any optimal solution for the problem of finding a MSCSS of $G \setminus \{v\}$. Since the vertex v is not a SAP in $G[E_{opt}]$, the graph $G[E_{opt}] \setminus \{v\}$ is strongly connected. Thus, we have $|E_{opt}| \geq |E'_{opt}|$. Consequently, by [ZNI03, Theorem 3], we have $|E'|/|E_{opt}| \leq |E'|/|E'_{opt}| \leq 5/3$. Since $|T_1 \cup T_2 \cup T_3^R \cup T_4^R| \leq 4(n-1)$ and $|E_{opt}| \geq n$, Algorithm 3.6.3 has approximation ratio $|E^*|/|E_{opt}| \leq 4 - 4/n + 5/3 = 17/3 - 4/n$. □

Theorem 3.6.6
Algorithm 3.6.3 runs in linear time.

Proof. The SAPs of G can be calculated in linear time using the algorithm of Italiano et al. [ILS12]. Two independent spanning trees of $G(v)$ can also be constructed in linear time [GT12b]. Furthermore, the algorithm of Zhao et al. [ZNI03] can be implemented in linear time. □

3.7. Approximation algorithm for the MS-2SBs problem

In this section we present an approximation algorithm for the MS-2SBs problem. Let $G = (V, E)$ be a strongly connected graph such that G is not 2-vertex-connected. Our algorithm consists of two main steps. The first step finds a SCSS G^* of G such that G^* and G have the same SAPs. The following lemma explains the purpose of this step.

Lemma 3.7.1
Let $G = (V, E)$ be a strongly connected graph and let S be the set of all SAPs of G. Let $G^* = (V, E^*)$ be a feasible solution for the MS-SAPs problem and let x, y be distinct vertices in V. Then for any vertex $z \in V \setminus (S \cup \{x, y\})$, the vertices x, y lie in the same SCC of $G^* \setminus \{z\}$.

Proof. Immediate from the definition of SAPs, since G and G^* have the same SAPs. □

Let x, y be two vertices in G such that $x \overset{2s}{\rightsquigarrow} y$ and let z be a vertex in $V \setminus \{x, y\}$ such that z is not SAP in G. The first step ensures that there exist at least one path from x to y and from y to x in $G^* \setminus \{z\}$. The second step

computes, for each SAP v of G, strongly connected spanning subgraphs of the subgraphs induced by the SCCs of $G \setminus \{v\}$.

Algorithm 3.7.2

Input: A strongly connected graph $G = (V, E)$.
Output: A SCSS of G with the same 2-strong blocks.

```
1   if G is 2-vertex-connected then
2     Run algorithm of Cheriyan and Thurimella [CT00] for minimum
3     cardinality 2-VCSS problem, improved in [Geo11].
4   else
5     lines 4–14 of Algorithm 3.6.3, giving G* = (V, E*).
6     for each SAP v of G do
7       Compute the SCCs of G \ {v}.
8       for each SCC C of G \ {v} do
9         if G*[C] is not strongly connected then
10          Find a SCSS (C, E') of G[C] with |E'| ≤ 2(|C| − 1).
11          E* ← E* ∪ E'.
12    Output G* = (V, E*).
```

In the following lemma we show that Algorithm 3.7.2 returns a feasible solution for the MS-2SBs problem.

Lemma 3.7.3
Let G^* be the output of Algorithm 3.7.2. Then G^*, G have the same 2-strong blocks and G^* is strongly connected.

Proof. Since each 2-vertex-connected graph is a 2-strong block, the algorithm of Cheriyan and Thurimella [CT00] returns a feasible solution when G is 2-vertex-connected. Let $G = (V, E)$ be a strongly connected graph such that G is not 2-vertex-connected. By Lemma 3.6.4, the directed graph G^* which is calculated in line 5 and the directed graph G have the same SAPs, and G^* is strongly connected. Therefore, the output G^* of Algorithm 3.7.2 and G have the same SAPs. Obviously, it is sufficient to show the following. Let $x, y \in V$ be distinct vertices such that $x \overset{2s}{\rightsquigarrow} y$ in G. We must show that $x \overset{2s}{\rightsquigarrow} y$ in the output G^* of Algorithm 3.7.2. Let $v \in V \setminus \{x, y\}$ be some vertex. By Lemma 3.7.1, we may assume that v is a SAP. Then x, y lie in the same SCC of $G \setminus \{v\}$.

The execution of the loop in lines 6–11 for v enforces that x, y are also in the same SCC of $G^* \setminus \{v\}$. □

Theorem 3.7.4
Algorithm 3.7.2 has an approximation factor of $(2t_{sap} + 17/3)$.

Proof. If G is 2-vertex-connected, the algorithm of Cheriyan and Thurimella [CT00] for the minimum cardinality 2-VCSS problem achieves an approximation ratio of 3/2. We consider now the case when G is not 2-vertex-connected. Let E_{opt} be an optimal solution for the MS-2SBs problem. The output G^* of Algorithm 3.7.2 consists of two edge sets E_1, E_2, where the edge set E_1 is computed in line 5 and the edge set E_2 is computed in lines 6–11. By Theorem 3.6.5, $|E_1|/|E_{opt}| \leq 17/3$. The number of iterations of the for-loop in lines 6–11 is t_{sap}. Because the SCCs of a directed graph are disjoint, we have $|E_2| < 2t_{sap}n$. Since $|E_{opt}| \geq n$, we have $|E_2|/|E_{opt}| < 2t_{sap}$. □

Theorem 3.7.5
Algorithm 3.7.2 runs in $O(m(\sqrt{n} + t_{sap}) + n^2)$ time.

Proof. The algorithm of Cheriyan and Thurimella [CT00] for the minimum cardinality 2-VCSS problem has running time $O(m^2)$. In 2011, Georgiadis [Geo11] improved it to $O(m\sqrt{n}+n^2)$. By Theorem 3.6.6, line 5 takes $O(n+m)$ time. The SCCs of a directed graph can be found in linear time using Tarjan's algorithm [Tar72]. Thus, lines 6–11 take $O(t_{sap}m)$ time. □

Notice that in lines 9–11 of Algorithm 3.7.2, every SCC C which does not contain any vertex of the 2-strong blocks of G can be safely disregarded.

3.8. Approximation algorithm for the MS-2EBs problem

In this section we present an approximation algorithm for the MS-2EBs problem. The idea of this algorithm (Algorithm 3.8.1) is similar to the idea of Algorithm 3.7.2.

Algorithm 3.8.1

Input: A strongly connected graph $G = (V, E)$.
Output: A SCSS of G with the same 2-edge blocks.

```
1   Choose a vertex v of G.
2   Compute two spanning trees T1, T2 of G(v) (rooted at v) such
3   that T1, T2 have only the edge dominators of G(v) in common.
4   Compute two spanning trees T3, T4 of G^R(v) (rooted at v) such
5   that T3, T4 have only the edge dominators of G^R(v) in common.
6   E* ← T1 ∪ T2 ∪ T3^R ∪ T4^R.
7   Find all the strong bridges in G.
8   for each strong bridge e of G do
9     Compute the SCCs of G \ {e}.
10    for each SCC C of G \ {e} do
11      if G*[C] is not strongly connected then
12        Find a SCSS (C, E') of G[C] with |E'| ≤ 2(|C| − 1).
13        E* ← E* ∪ E'.
14  Output G* = (V, E*).
```

Lemma 3.8.2
Let G^* be the output of Algorithm 3.8.1. Then G^* is strongly connected and the directed graphs G^*, G have the same strong bridges.

Proof. Since $T_1 \cup T_3^R \subseteq E^*$, the graph G^* is strongly connected. Tarjan [Tar76] proved that there exist two spanning trees (rooted at v) of $G(v)$ that have only the edge dominators of $G(v)$ in common and he gave algorithms for computing them. Italiano et al. [ILS12] showed that edge $e \in E$ is strong bridge if and only if e is an edge dominator in $G(v)$ or in $G^R(v)$. Therefore, the directed graphs $G, (V, T_1 \cup T_2 \cup T_3^R \cup T_4^R)$ have the same strong bridges. Since $(V, T_1 \cup T_2 \cup T_3^R \cup T_4^R)$ is a subgraph of G^* and G^* is a subgraph of G, the directed graphs G^*, G have the same strong bridges. □

Next we show that Algorithm 3.8.1 outputs a feasible solution for the MS-2EBs problem.

Lemma 3.8.3
Let $G^* = (V, E^*)$ be the output of Algorithm 3.8.1. Then G^*, G have the same 2-edge blocks.

Proof. Let x, y be distinct vertices such that $x \overset{2e}{\rightsquigarrow} y$ in G. We must show that $x \overset{2e}{\rightsquigarrow} y$ in G^*. By Lemma 3.3.1, we need to show that x, y lie in the same SCC of $G^* \setminus \{e\}$ for any edge $e \in E^*$. Let e be an edge in G^*. We consider two cases.

1. e is not a strong bridge in G^*. By Lemma 3.8.2, G^* is strongly connected. Hence, by definition of strong bridges, the vertices x, y lie in the same SCC of $G^* \setminus \{e\}$.

2. e is a strong bridge in G^*. By Lemma 3.8.2, G, G^* have the same strong bridges. Since x, y lie in the same SCC of $G \setminus \{e\}$, the execution of the loop in lines 8–13 enforces that the vertices x, y are also in the same SCC of $G^* \setminus \{e\}$.

□

Theorem 3.8.4
Let $G^* = (V, E^*)$ be the output of Algorithm 3.8.1. Then $|E^*| < (4+2t_{sb})n$.

Proof. Let E_1 be the edge set which is computed in lines 8–13. Notice that the SCCs of a directed graph are disjoint and we add at most $2(|C| - 1)$ edges for each SCC C in lines 12–13. Since the number of iterations of the for-loop in lines 8–13 is t_{sb}, we have $|E_1| \leq 2t_{sb}(n-1)$. Therefore, we have $|E^*| = |T_1 \cup T_2 \cup T_3^R \cup T_4^R| + |E_1| \leq 4(n-1) + 2t_{sb}(n-1) < (4 + 2t_{sb})n$. □

Let G be a strongly connected graph. Since each SCSS of G has at least n edges, Algorithm 3.8.1 achieves an approximation ratio of $(4 + 2t_{sb})$.

Theorem 3.8.5
The running time of Algorithm 3.8.1 is $O((t_{sb} + \alpha(n, m))m)$.

Proof. Two spanning trees T_1, T_2 of $G(v)$ (rooted at v) such that T_1, T_2 have only the edge dominators of $G(v)$ in common can be computed in $O(m\alpha(n, m))$ time by using Tarjan's algorithm [Tar76], where $\alpha(n, m)$ is a very slowly function related to a functional inverse of Ackermann's function. The strong bridges of a strongly connected graph can be computed in linear time using the algorithm

of Italiano et al. [ILS12]. Moreover, the number of iterations of the for-loop in lines 6–13 is t_{sb}. The total time for Algorithm 3.8.1 is therefore $O((t_{sb} + \alpha(n, m))m)$. □

Notice that by Lemma 3.4.1, we can obtain a $(2(t_{sap} + t_{sb}) + 29/3)$ approximation algorithm for the MS-2DBs problem by combining Algorithm 3.7.2 and Algorithm 3.8.1. This algorithm whose running time is $O((t_{sap} + t_{sb} + \sqrt{n} + \alpha(n, m))m + n^2)$ might be useful when $t_{sap} + t_{sb}$ is small.

3.9. Open problems

The *k-strong blocks* of directed graphs, which are natural generalization of 2-strong blocks, are similar to the k-blocks of undirected graphs [Car+14]. Let $G = (V, E)$ be a directed graph. We define a relation $\overset{ks}{\leftrightsquigarrow}$ as follows. For any pair of distinct vertices $x, y \in V$, we write $x \overset{ks}{\leftrightsquigarrow} y$ if for each subset $X \subseteq V \setminus \{x, y\}$ with $|X| < k$, the vertices x and y lie in the same SCC of $G \setminus X$. A *k-strong block* in a directed graph G is a maximal vertex set $C^{ks} \subseteq V$ with $|C^{ks}| \geq k$ such that for each pair of distinct vertices $x, y \in C^{ks}$, we have $x \overset{ks}{\leftrightsquigarrow} y$. One can show that any two k-strong blocks have at most $k - 1$ vertices in common. A simple algorithm was given in [Car+14] to find the k-blocks of an undirected graph for any fixed k. We noticed that this algorithm is also able to compute the k-strong blocks of a directed graph in $O(\min\{k, \sqrt{n}\} \cdot m \cdot n^2)$-time. We just need to modify the pre-processing step and the definition of separation. Let $G = (V, E)$ be a directed graph. An ordered pair (C, D) such that $C, D \subseteq V$ and $C \cup D = V$ is a *separation* of G if there is no edge from $C \setminus D$ to $D \setminus C$ or there is no edge from $D \setminus C$ to $C \setminus D$. In the pre-processing step we construct a new undirected graph $H_k = (V, E_k)$ as follows. For each pair of distinct vertices x, y of G, if $x \overset{ks}{\leftrightsquigarrow} y$ in G, we add an undirected edge (x, y) to E_k (see Lemma 3.9.1 and Lemma 3.9.2). Furthermore, for every pair (x, y) of H_k with $(x, y) \notin E_k$, we label it with some separation (C, D) such that $|C \cap D| < k$ and $x \in C, y \in D$.

Lemma 3.9.1
Let $G = (V, E)$ be a directed graph and let x, y be distinct vertices of G. Then $x \overset{ks}{\leftrightsquigarrow} y$ if and only if x, y satisfy one of the following conditions.

1. $\{(x, y), (y, x)\} \subseteq E$.

2. $(x, y) \in E, (y, x) \notin E$ and there are k-vertex-disjoint paths from y to x in G.

3. $(y, x) \in E, (x, y) \notin E$ and there are k-vertex-disjoint paths from x to y in G.

4. $\{(x, y), (y, x)\} \cap E = \emptyset$ and there exist k-vertex-disjoint paths from x to y and from y to x in G.

Proof. This follows immediately from Menger's Theorem for vertex connectivity. □

Lemma 3.9.2
Let (C, D) be a separation of a directed graph $G = (V, E)$ such that $|C \cap D| < k$. Then $C \setminus D$ and $D \setminus C$ do not lie in the same k-strong block of G.

Proof. Let x be any vertex in $C \setminus D$ and let y be any vertex in $D \setminus C$. By the definition of separation, there is either no path from x to y or no path from y to x in $G \setminus (C \cap D)$. Thus, x, y do not lie in the same SCC of $G \setminus (C \cap D)$. □

We leave as an open problem whether there exists any approximation algorithm for the problem of finding MSCSS with same k-strong blocks of a strongly connected graph for $k > 2$. Another open question is whether there is an approximation algorithm for the problem of finding MSCSS with the same k-directed blocks when $k > 2$.

It is also possible to generalize the MS-2EBs problem. Let $G = (V, E)$ be a strongly connected graph. We define a relation $\overset{ke}{\leftrightsquigarrow}$ as follows. For any pair of distinct vertices $x, y \in V$, we write $x \overset{ke}{\leftrightsquigarrow} y$ if for each edge subset $Y \subseteq E$ with $|Y| < k$, the vertices x, y lie in the same SCC of $G \setminus Y$. A *k-edge blocks* of G is a maximal vertex subset $C^{ke} \subseteq V$ with $|C^{ke}| \geq k$ such that for each pair of vertices $x, y \in C^{ke}$ we have $x \overset{ke}{\leftrightsquigarrow} y$.

Lemma 3.9.3
The k-edge blocks of a strongly connected graph are disjoint.

Proof. The proof is similar to our proof of Lemma 3.3.2. □

The k-edge blocks of a directed graph can be found in $O(n^3m)$ time using maximum flow algorithms [Orl13; GT14] since the relation $\overset{ke}{\leftrightsquigarrow}$ is symmetric and transitive.

4. Removing SAPs, non-SAPs or strong bridges to change 2-blocks

In this chapter we study several problems related to the effect of removing SAPs, non-SAPs, and strong bridges on 2-blocks in directed graphs. Let $G = (V, E)$ be a strongly connected graph. Section 4.1 deals with the problem of finding a minimum size subset E^* of E^{sb} such that the removal of E^* has the same effect on the 2-edge blocks of G as the removal of E^{sb}. Deleting the vertices that are not SAPs of G and do not lie in any 2-directed block of G may change the 2-directed blocks of G. This will be considered in Section 4.2. Let $C_1^{2d}, C_2^{2d}, \ldots, C_l^{2d}$ be the 2-directed blocks of G. In Section 4.3 we study the problem of finding a minimum cardinality subset of $V^{sap} \setminus (\bigcup_{1 \leq i \leq l} C_i^{2d})$ whose removal from G leaves a directed graph containing the same 2-directed blocks as $G \setminus (V^{sap} \setminus (\bigcup_{1 \leq i \leq l} C_i^{2d}))$. In Section 4.4 we discuss the problem of finding a minimum size subset of strong bridges whose deletion destroys a subset of a 2-edge block. Let $C_1^{2s}, C_2^{2s}, \ldots, C_t^{2s}$ be the 2-strong blocks of G. Section 4.5 treats the problem of finding a minimum cardinality subset $V^* \subseteq V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2s})$ such that the graphs $G \setminus V^*$ and $G \setminus (V^{sap} \setminus (\bigcup_{1 \leq i \leq t}))$ have the same 2-strong blocks. We consider the problem of finding a minimum size subset of non-SAPs whose removal changes some 2-strong blocks in Section 4.6. Moreover, we study problems related to the strong bridges that disrupt the 2-edge blocks of G in Section 4.7.

4.1. Removing strong bridges to change 2-edge blocks

In this section we study the following optimization problem.

Problem 4.1.1
Input: A strongly connected graph $G = (V, E)$.
Task: Find a minimum size subset $E^* \subseteq E^{sb}$ such that the 2-edge blocks of $G \setminus E^*$ coincide with the 2-edge blocks of $G \setminus E^{sb}$.

Consider the strongly connected graph G of Figure 1.5. If its strong bridges are removed, then the 2-edge blocks of the remaining graph are $C_1^{2e} = \{1, 14, 15\}$ and $C_2^{2e} = \{10, 11, 12, 13\}$. Since the 2-edge blocks of the directed graph $G \setminus \{(3, 7), (3, 6)\}$ are also C_1^{2e}, C_2^{2e} and the edges $(3, 7), (3, 6)$ are strong bridges in G, the set $\{(3, 7), (3, 6)\}$ is a feasible solution for Problem 4.1.1 on the graph G.

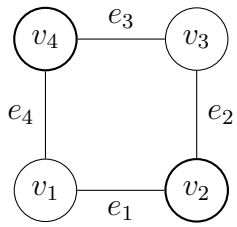
We prove that Problem 4.1.1 is NP-hard by reducing the vertex cover problem to it. The decision version of the vertex cover problem is defined as follows. Given an undirected graph $G = (V, E)$ and an integer l. Is there a vertex set $U \subseteq V$ with $|U| \leq l$ such that for each edge $(v, w) \in E$ we have $\{v, w\} \cap U \neq \emptyset$? This problem is NP-complete [Kar72].

The decision version of Problem 4.1.1 can be formulated as follows. Given a strongly connected graph $G = (V, E)$ and an integer r. Does there exist a subset $E^* \subseteq E^{sb}$ with $|E^*| \leq r$ such that the directed graph $G \setminus E^*$ has the same 2-edge blocks as the directed graph $G \setminus E^{sb}$?

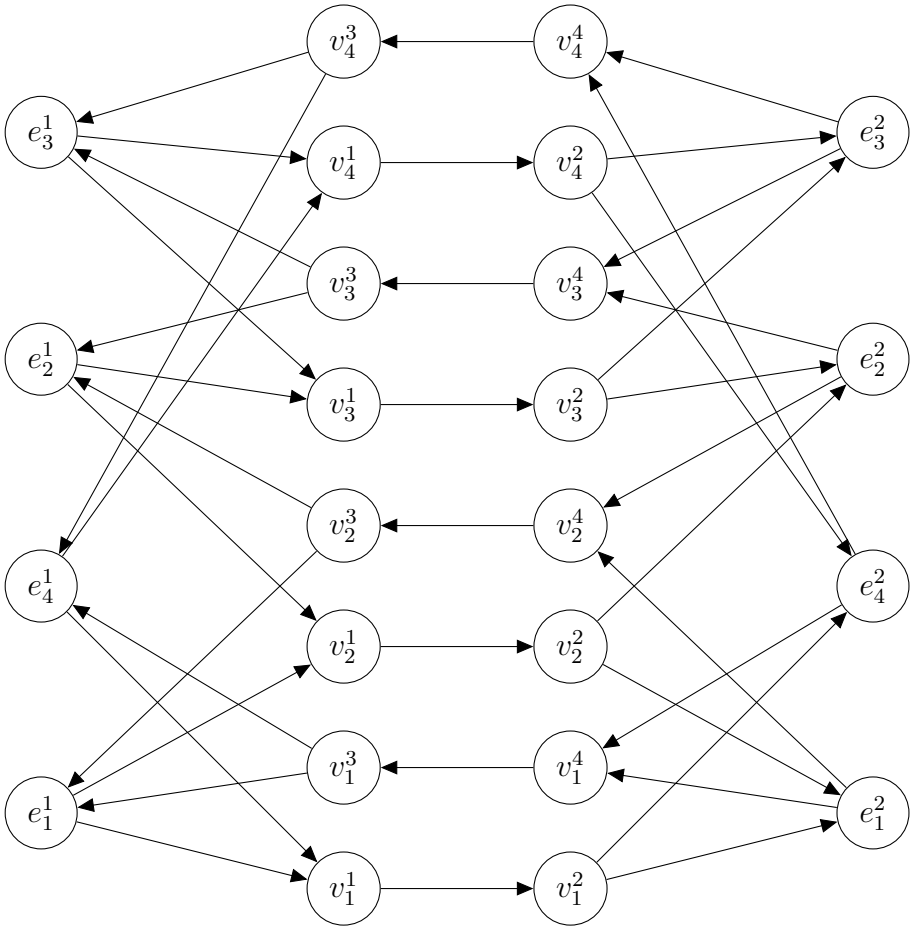
Lemma 4.1.2
The decision version of Problem 4.1.1 is NP-complete.

Proof. It is clear that Problem 4.1.1 belongs to NP. Let $(G = (V, E), l)$ be an instance of the vertex cover problem. We may assume that G is connected. We construct an instance $(G' = (V', E'), r)$ of Problem 4.1.1 as follows. For every vertex $v \in V$, we add four new vertices v^1, v^2, v^3, v^4 to V' and two directed edges $(v^1, v^2), (v^4, v^3)$ to E'. For every undirected edge $e = (u, w) \in E$, we add two new vertices e^1, e^2 to V' and four directed edges $(e^1, u^1), (e^1, w^1), (u^3, e^1), (w^3, e^1), (e^2, u^4), (e^2, w^4), (u^2, e^2), (w^2, e^2)$ to the set E'. An example is given in Figure 4.1.

For every edge $e = (u, w) \in E$, by construction the vertices of G' that correspond to the edge e and the vertices u, w lie in the same SCC of G. Consequently, G' is strongly connected. The strongly connected graph G' can be constructed in polynomial time since G' has $4|V|+2|E|$ vertices and $2|V|+4|E|$ edges.



(a) $G = (V, E)$, $U = \{v_2, v_4\}$



(b) $G' = (V', E'), E^* = \{(v_2^1, v_2^2), (v_4^1, v_4^2)\}$

Figure 4.1.: Reducing the vertex cover problem to Problem 4.1.1

Let w^1, w^2, w^3, w^4 be the vertices in G' which correspond to a vertex $w \in V$. Since there is no path from w^1 to w^2 in $G' \setminus \{(w^1, w^2)\}$, the directed graph $G' \setminus \{(w^1, w^2)\}$ is not strongly connected. Therefore, the edge (w^1, w^2) is a strong bridge of G'. Moreover, the directed graph $G' \setminus \{(w^4, w^3)\}$ is not strongly connected because the vertex w^3 is not reachable from w^4 in this graph. Consequently, the edge (w^4, w^3) is also a strong bridge in G'. Therefore, all the edges of G' that correspond to the vertices of G are strong bridges.

We now prove that only the vertices in G' that correspond to the edges of G lie in the 2-edge blocks of G'. For every pair of vertices e^1, e^2 which correspond to an undirected edge $e = (u, w)$ in G, there are two edge-disjoint paths from e^1 to e^2 in G', namely (e^1, u^1, u^2, e^2) and (e^1, w^1, w^2, e^2). Moreover, there exist two edge-disjoint paths from e^2 to e^1 in G', namely (e^2, u^4, u^3, e^1) and (e^2, w^4, w^3, e^1). Hence, we have $e^1 \overset{2e}{\leftrightsquigarrow} e^2$. Let v be a vertex in G. The vertices v^1, v^4 have outdegree 1 and the vertices v^2, v^3 have indegree 1 in G', where v^1, v^2, v^3, v^4 correspond to the vertex v. Therefore, the vertices v^1, v^2, v^3, v^4 do not lie in any 2-edge block of G'.

Note that removing the edges of G' that correspond to the vertices of G destroys all the 2-edge blocks of G'. Since these edges are strong bridges in G', the removal of E^{sb} from G' also destroys all the 2-edge blocks of G'. Thus, it is sufficient to show the following.

Claim 4.1.3
The graph G has a vertex cover of size at most l if and only if the strongly connected graph G' contains a subset $E^* \subseteq E^{sb}$ of size at most $r = l$ such that the directed graph $G' \setminus E^*$ has no 2-edge blocks.

Proof. "⇒": Let $U \subseteq V$ be a vertex cover of G such that $|U| \leq l$. Let $E^* = \{(v^1, v^2) \mid (v^1, v^2) \in E' \text{ and } v \in U\}$. As we have seen, all the edges of this set are strong bridges of G'. Obviously, the set E^* contains at most l edges. Let $e = (u, w)$ be an undirected edge in G. We assume without of loss of generality that $w \in U$. Then E^* contains (w^1, w^2). We consider the vertices $e^1, e^2 \in V'$ that correspond to the edge $e \in E$. Let C^{2e} be the 2-edge block of G' that contains the vertices e^1, e^2. Let $x \in C^{2e} \setminus \{e^1\}$. There are two cases to consider:

1. $(u^1, u^2) \notin E^*$. Then all paths from e^1 to x contain the edge (u^1, u^2) in the graph $G' \setminus E^*$ since $(w^1, w^2) \in E^*$. Thus, by Menger's Theorem for edge-connectivity, there are no two edge-disjoint paths from the vertex e^1 to the vertex x in $G' \setminus E^*$.

2. $(u^1, u^2) \in E^*$. In this case the vertex x is not reachable from e^1 in $G' \setminus E^*$.

For every vertex $y \in C^{2e} \setminus e^2$ there do not exist two edge-disjoint paths from y to e^2 in $G \setminus E^*$. Since the 2-edge blocks of G' contain only the vertices of G' that correspond to the edges of G, deleting the set E^* from G' destroys the 2-edge block C^{2e}.

"$\Leftarrow$": Let E^* be a subset of E^{sb} with $|E^*| \leq l$ such that $G' \setminus E^*$ does not contain any 2-edge block. Let $U = \{v \mid \{(e^1, v^1), (v^1, v^2), (v^2, e^2), (e^2, v^4), (v^4, v^3), (v^3, e^1)\} \cap E^* \neq \emptyset$ and $v \in V\}$. Clearly, $|U| \leq l$. Assume for a contradiction that U is not a vertex cover of G. Then there is an edge $e = (u, w)$ in G such that $\{u, w\} \cap U = \emptyset$. This implies that $\{(e^1, u^1), (e^1, w^1), (u^3, e^1), (w^3, e^1), (u^1, u^2), (u^4, u^3), (w^1, w^2), (w^4, w^3), (e^2, u^4), (e^2, w^4), (u^2, e^2), (w^2, e^2)\} \cap E^* \doteq \emptyset$. Therefore, in the graph $G' \setminus E^*$, there exist two edge-disjoint paths from e^1 to e^2, namely (e^1, u^1, u^2, e^2) and (e^1, w^1, w^2, e^2), and there are two edge-disjoint paths from e^2 to e^1, namely (e^2, u^4, u^3, e^1) and (e^2, w^4, w^3, e^1). Hence, the directed graph $G' \setminus E^*$ has a 2-edge block containing at least the vertices e^1, e^2, a contradiction. □

Now we prove that Problem 4.1.1 is solvable in polynomial time for strongly connected graphs that contain only one 2-edge block which is of size 2.

Lemma 4.1.4
Let $G = (V, E)$ be a strongly connected graph such that there is only one 2-edge block C^{2e} in G and C^{2e} contains only two vertices x, y. Let E^{opt} be an optimal solution for Problem 4.1.1 on G. Then there are a path from x to y and a path from y to x in $G \setminus E^{opt}$.

Proof. We consider two cases:

1. The set C^{2e} is a 2-edge block of $G \setminus E^{sb}$. In this case $E^{opt} = \emptyset$. Therefore, the vertices x and y are in the same SCC of $G \setminus E^{opt}$.

2. The set C^{2e} is not a 2-edge block of $G \setminus E^{sb}$. In this case $E^{opt} \neq \emptyset$. Without loss of generality, assume that in $G \setminus E^{opt}$ there is no path from x to y. Let e be an edge of E^{opt}. Since E^{opt} is an optimal solution, $E^{opt} \setminus \{e\}$ is not a solution for Problem 4.1.1 on G. Therefore, there are at least two edge-disjoint paths from x to y in $G \setminus (E^{opt} \setminus \{e\}) = (V, (E \setminus E^{opt}) \cup \{e\})$. By Menger's theorem for edge connectivity, the removal of e from $(E \setminus E^{opt}) \cup \{e\}$ does not destroy all paths from x to y, so in $G \setminus E^{opt}$ there exists a path from x to y, a contradiction. □

Let $G = (V, E)$ be a strongly connected graph such that there is only one 2-edge block C^{2e} in G and the size of C^{2e} is 2. We define a capacity function $w : E \to \mathbb{N}$ as follows:

$$w(e) = \begin{cases} 1 & e \text{ is a strong bridge of } G \\ 2m & \text{otherwise} \end{cases}$$

Let x, y be a pair of distinct vertices in G. An $x-y$-cut is a set of edges whose removal from G destroys all the paths from x to y. It is well known that a minimum capacity $x-y$-cut can be computed by maximum flow algorithms (e.g. see [GT14]).

Lemma 4.1.5
Let $G = (V, E)$ be a strongly connected graph such that there is only one 2-edge block $C^{2e} = \{x, y\}$ in G and there is no 2-edge block in the graph obtained from G by deleting all the strong bridges of G. Let E^{opt} be an optimal solution for Problem 4.1.1 on G. Then there is an edge $e \in E \setminus E^{opt}$ such that each minimum $x-y$-cut in $G \setminus \{e\}$ or each minimum $y-x$-cut is an optimal solution for Problem 4.1.1 on G.

Proof. Because there does not exist a 2-edge block in the graph obtained from G by removing its strong bridges, we have $E^{opt} \neq \emptyset$. We consider two cases:

1. There do not exist two edge-disjoint paths from x to y in $G \setminus E^{opt}$. By Lemma 4.1.4, the vertex y is reachable from vertex x in $G \setminus E^{opt}$. By Menger's Theorem for edge connectivity, there is an edge $e \in E \setminus E^{opt}$ that lies on each path from x to y in $G \setminus E^{opt}$. Since there is no path from x to y in $G \setminus (\{e\} \cup E^{opt})$, E^{opt} is a set of edges whose removal destroys all paths from x to y in $G \setminus \{e\}$, i.e. E^{opt} is a $x-y$-cut in $G \setminus \{e\}$. The capacity of the $x-y$-cut E^{opt} is $|E^{opt}|$ because this cut contains only strong bridges. Let C_{xy} be a minimum $x-y$-cut in $G \setminus \{e\}$. Then the capacity of C_{xy} is less than or equal to $|E^{opt}|$. Since each edge which is not a strong bridge has capacity $2m$, all the edges of C_{xy} are strong bridges. Thus, the capacity of C_{xy} is $|C_{xy}|$. Since C_{xy} is a feasible solution for Problem 4.1.1 on G and $|C_{xy}| \leq |E^{opt}|$, we have $|E^{opt}| = |C_{xy}|$. Thus, E^{opt} is also a minimum $x-y$-cut in $G \setminus \{e\}$ and C_{xy} is an optimal solution.

2. There are no 2-edge-disjoint paths from y to x in $G \setminus E^{opt}$. In this case each minimum $y-x$-cut is an optimal solution. The proof of this case is similar to case 1. □

Algorithm 4.1.6

Input: A strongly connected graph $G = (V, E)$ containing only 2-edge block $C^{2e} = \{x, y\}$.
Output: An optimal solution E^{opt} for Problem 4.1.1 on G.

```
1   Compute E^sb.
2   if C^2e is a 2-edge block of G \ E^sb then
3     E^opt ← ∅.
4   else
5     E^opt ← E^sb.
6     for each edge e ∈ E do
7       find a minimum x − y-cut C_xy and a minimum y − x-cut C_yx in G \ {e}.
8       if |C_xy| < |E^opt| then
9         E^opt ← C_xy.
10      if |C_yx| < |E^opt| then
11        E^opt ← C_yx.
12    Output E^opt.
```

Lemma 4.1.7
Algorithm 4.1.6 computes an optimal solution for Problem 4.1.1 for strongly connected graphs G such that G contains only one 2-edge block C^{2e} and its cardinality is 2.

Proof. It suffices to consider the case when removing E^{sb} from G destroys C^{2s}, i.e., $E^{opt} \neq \emptyset$. First we prove that the output of Algorithm 4.1.6 is a feasible solution. We consider two cases:

1) $E^{opt} = E^{sb}$. Obviously, E^{opt} is a feasible solution.

2) $|E^{opt}| < |E^{sb}|$. Assume without loss of generality that $E^{opt} = C_{xy}$, where C_{xy} is a minimum $x - y$-cut in $G \setminus \{e\}$ for some edge $e \in E$. Then all the edges in E^{opt} are strong bridges. Moreover, the edge e lies on each path from x to y in $G \setminus E^{opt}$. Thus, by Menger's Theorem for edge connectivity, there do not exist two edge-disjoint paths from x to y in $G \setminus E^{opt}$. Therefore, the output E^{opt} is a feasible solution.

From Lemma 4.1.5 it follows that the output E^{opt} is an optimal solution. □

Theorem 4.1.8
Algorithm 4.1.6 runs in $O(nm^2)$ time.

Proof. The strong bridge of G can be computed in linear time using the algorithm of Italiano et al. [ILS12]. The 2-edge blocks of G can be calculated in linear time using the algorithm of Georgiadis et al [Geo+15a]. Thus, lines 2–3 take $O(m)$ time. A minimum $x - y$-cut can be calculated in $O(nm)$ time by maximum flow algorithms [Orl13; GT14]. The number of iterations of the for loop is m. Therefore, the running time of algorithm 4.1.6 is $O(nm^2)$. □

4.2. Removing non-SAPs to change 2-directed blocks

Let $G = (V, E)$ be a strongly connected graph. If we remove the vertices of $V \setminus V^{sap}$ that do not lie in the 2-vccs of G, the 2-vccs do not change. Unlike the 2-vccs, the 2-directed blocks of G may be changed by deleting the vertices of $V \setminus V^{sap}$ which do not belong to these 2-directed blocks (see Figure 4.2).

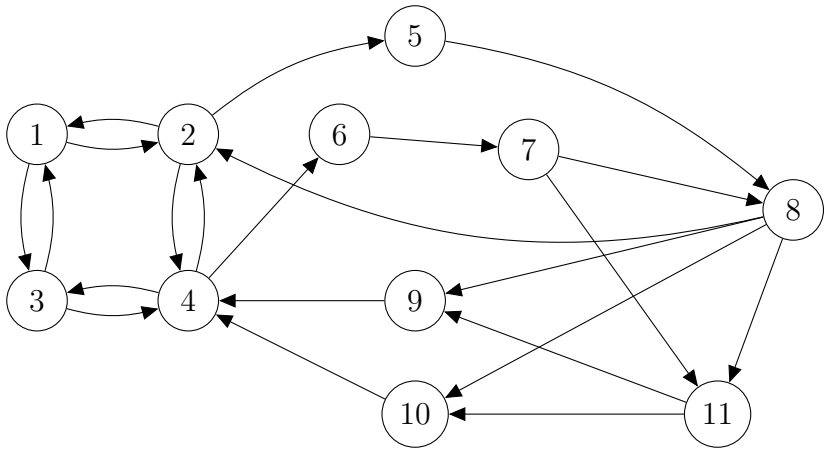


Figure 4.2.: A strongly connected graph $G = (V, E)$. This graph has two 2-directed blocks $C_1^{2d} = \{1, 2, 3, 4, 8\}$ and $C_2^{2d} = \{4, 11\}$. The graph obtained by removing $V \setminus (V^{sap} \cup C_1^{2d} \cup C_2^{2d}) = \{5, 9, 10\}$ contains only the 2-directed block $C^{2d} = \{1, 2, 3, 4\}$.

In this section we deal with the following problem.

Problem 4.2.1
Input: A strongly connected graph $G = (V, E)$ with the 2-directed blocks $C_1^{2d}, C_2^{2d}, \ldots, C_t^{2d}$.
Task: Find a vertex subset $V^* \subseteq V \setminus (V^{sap} \cup (\bigcup_{1 \leq i \leq t} C_i^{2d}))$ of minimum cardinality such that the removal of V^* from G has the same influence on the 2-directed blocks of G as the removal of $V \setminus (V^{sap} \cup (\bigcup_{1 \leq i \leq t} C_i^{2d}))$ from the graph G.

We show that Problem 4.2.1 is NP-hard by reducing the vertex cover problem to it.

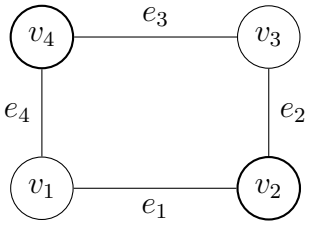
We define the decision version of Problem 4.2.1 as follows. Given a strongly connected graph $G = (V, E)$ and an integer r. Does there exist a subset $V^* \subseteq V \setminus (V^{sap} \cup (\bigcup_{1 \leq i \leq t} C_i^{2d}))$ of size at most r such that the 2-directed blocks of $G \setminus V^*$ coincide with the 2-directed blocks of $G \setminus V_1$, where $C_1^{2d}, C_2^{2d}, \ldots, C_t^{2d}$ are the 2-directed blocks of G and V_1 is the set of vertices that are not SAPs of G and are not elements of the set $\bigcup_{1 \leq i \leq t} C_i^{2d}$, i.e, $V_1 = V \setminus (V^{sap} \cup (\bigcup_{1 \leq i \leq t} C_i^{2d}))$?

Lemma 4.2.2
The decision version of Problem 4.2.1 is NP-complete.

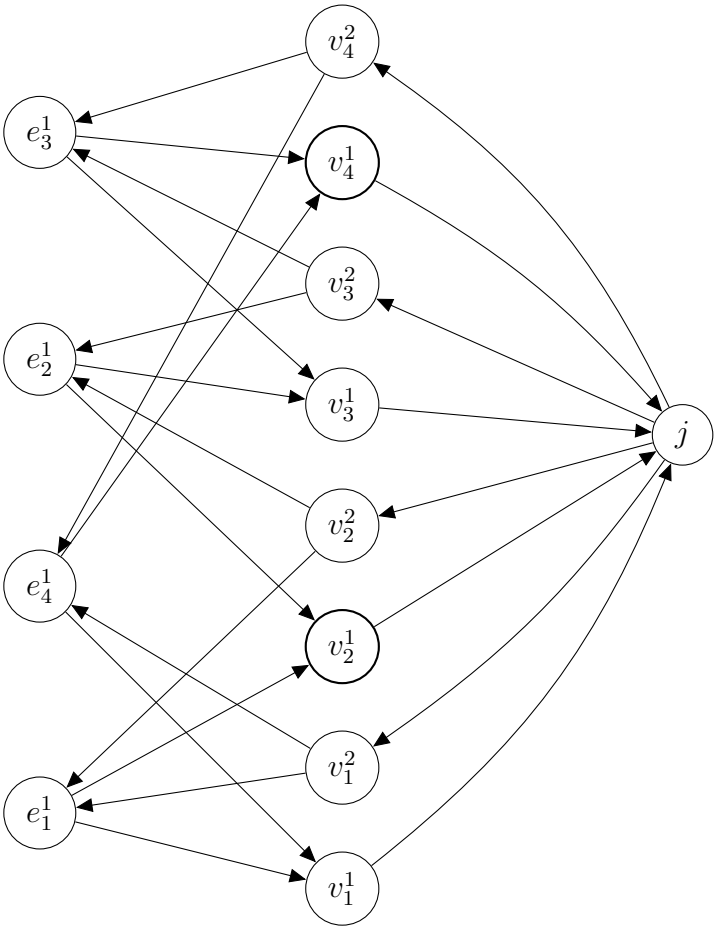
Proof. It is obvious that Problem 4.2.1 is in NP. Let $(G = (V, E), l)$ be an instance of the vertex cover problem. We construct an instance $(G' = (V', E'), r)$ of Problem 4.2.1 as follows. We add a new vertex j to G'. Furthermore, for each vertex $v \in V$, we add two new vertices v^1, v^2 to V' and two new directed edges $(v^1, j), (j, v^2)$ to E'. For each undirected edge $e = (w, u)$ in G we add one new vertex e^1 to V' and four directed edges $(e^1, w^1), (e^1, u^1), (w^2, e^1), (u^2, e^1)$. Figure 4.3 shows an example for this construction. The directed graph G' can be constructed in polynomial time because $|V'| = |E|+2|V|+1$ and $|E'| = 4|E|+2|V|$.

For each vertex $v' \in V' \setminus \{j\}$, there exist a path from j to v' and a path from v' to j in G'. Therefore, the directed graph $G' = (V', E')$ is strongly connected. It is also easy to see that all vertices of G' that correspond to the vertices of G are not SAPs.

We now prove that for each vertex $e^1 \in V'$ that corresponds to undirected edge $e \in E$, the set $\{e^1, j\}$ is a 2-directed block of G'. For each vertex e^1 of G' which corresponds to undirected edge $e = (u, w) \in E$, there exist two vertex-disjoint paths (e^1, u^1, j) and (e^1, w^1, j) from e^1 to j in G'. Moreover, there are two vertex-disjoint paths (j, u^2, e^1) and (j, w^2, e^1) from j to e^1 in G'.



(a) $G = (V, E)$, $U = \{v_2, v_4\}$



(b) $G' = (V', E'), V^* = \{v_2^1, v_4^1\}$

Figure 4.3.: Reducing the vertex cover problem to Problem 4.2.1

Thus, we have $e^1 \overset{2}{\rightsquigarrow} j$. Let y_1^1, y_2^1 be two distinct vertices in G' such that y_1^1 corresponds to undirected edge $y_1 \in E$ and y_2^1 corresponds to undirected edge $y_2 \in E$. Since in $G^{'}$ each path from y_1^1 to y_2^1 contains the vertex j, the vertices y_1^1, y_2^1 do not lie in the same SCC of $G' \setminus \{j\}$. Therefore, by Lemma 3.1.1 , the vertices y_1^1, y_2^1 lie in distinct 2-directed blocks of G'. For every vertex $v \in V$, the vertices v^1, v^2 of G' that correspond to v do not lie in any 2-directed block of G' since v^1 has outdegree 1 and v^2 has indegree 1 in G'. Observe that $G' \setminus (V' \setminus (\bigcup_{1 \leq i \leq |E|} \{e_i^1, j\}))$ has no 2-directed blocks and none of the vertices of $V' \setminus (\bigcup_{1 \leq i \leq |E|} \{e_i^1, j\})$ is a SAP of G'.

Since the sets $\{e_i^1, j\}$, for $e \in E$, are the 2-directed blocks of G' and the directed graph $G' \setminus (V' \setminus (\bigcup_{1 \leq i \leq |E|} \{e_i^1, j\}))$ does not contain any 2-directed blocks, it suffices to show the following.

Claim 4.2.3
There exists a vertex cover of size at most l in the graph G if and only if there exists a subset $V^* \subseteq V' \setminus (\bigcup_{1 \leq i \leq |E|} \{e_i^1, j\})$ of size at most $r = l$ such that $G' \setminus V^*$ has no 2-directed blocks.

Proof. "⇒": Let U be a vertex cover of G such that $|U| \leq l$. Let $V^* = \{v^1 \mid v^1 \in V' \text{ and } v \in U\}$. Clearly, $|V^*| = |U|$. Let C^{2d} be any 2-directed block of G'. Then $C^{2d} = \{e^1, j\}$ for some vertex e^1 corresponding to undirected edge $e \in E$. Since U is a vertex cover of G, we have $U \cap \{u, w\} \neq \emptyset$. There are two cases to consider:

1. $\{u, w\} \subseteq U$. Then $\{u^1, w^1\}$ is a subset of V^*. Thus, there is no path from e^1 to j in $G' \setminus V^*$.

2. Either u or w belongs to U. Assume without loss of generality that $w \in U$. Then w^1 belongs to V^*. Therefore, each path from e^1 to j contains the vertex u^1. Moreover, we have $(e^1, j) \notin E'$. By Menger's Theorem for vertex-connectivity there do not exist two vertex-disjoint paths from e^1 to j in $G' \setminus V^*$.

In both cases the vertices e^1, j can not be in the same 2-directed block of G'. As a consequence, the directed graph $G' \setminus V^*$ does not contain any 2-directed block.

"⇐": Let V^* be a subset of $V' \setminus (\bigcup_{1 \leq i \leq |E|} \{e_i^1, j\})$ of size at most l such that $G' \setminus V^*$ does not have any 2-directed block. Then we have $V^* \subseteq (\bigcup_{v \in V} \{v^1, v^2\})$. Let $U = \{v \mid \{v^1, v^2\} \cap V^* \neq \emptyset \text{ and } v \in V\}$. It is easy to see that $|U| \leq l$. Assume

for a contradiction that U is not a vertex cover of G. Then there is an edge $e = (u, w) \in E$ such that $\{u, w\} \cap U = \emptyset$. This implies that $\{u^1, u^2, w^1, w^2\} \cap V^* = \emptyset$. Therefore, there exist two vertex-disjoint paths $(e^1, u^1, j), (e^1, w^1, j)$ from e^1 to j and two vertex-disjoint paths $(j, u^2, e^1), (j, w^2, e^1)$ from j to e^1 in $G' \setminus V^*$. Because the vertices e^1, j belong to the same 2-directed block of $G' \setminus V^*$, there exists a 2-directed block containing at least the vertices e^1, j in the graph $G' \setminus V^*$, a contradiction. □

4.3. Removing SAPs to change 2-directed blocks

Consider the strongly connected graph G shown in Figure 1.4. Note that deleting the SAPs of G that do not lie in the 2-directed blocks of G changes these 2-directed blocks. We want to find a minimum size subset of V^{sap} that do not lie in the 2-directed blocks of G and has the same effect. We can formulate this problem as follows.

Problem 4.3.1
Input: A strongly connected graph $G = (V, E)$ whose 2-directed blocks are $C_1^{2d}, C_2^{2d}, \ldots, C_t^{2d}$.
Task: Find a minimum cardinality vertex subset $V^* \subseteq V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2d})$ such that the directed graph $G \setminus (V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2d}))$ has the same 2-directed blocks as $G \setminus V^*$.

The decision version of Problem 4.3.1 can be defined as follows. Given a strongly connected graph $G = (V, E)$ and an integer r. Is there a subset $V^* \subseteq V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2d})$ of size at most r such that the 2-directed blocks of $G \setminus (V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2d}))$ coincide with the 2-directed blocks of $G \setminus V^*$, where $C_1^{2d}, C_2^{2d}, \ldots, C_t^{2d}$ are the 2-directed blocks of G?

Lemma 4.3.2
The decision version of Problem 4.3.1 is NP-complete.

Proof. We reduce the vertex cover problem to this problem. Clearly, Problem 4.3.1 is in NP. Suppose that $(G = (V, E), l)$ is an input of the vertex cover problem. An input $(G' = (V', E'), r)$ of Problem 4.3.1 can be built in the following way. For each vertex $v \in V$, the directed graph G' contains four vertices v^1, v^2, v^3, v^4 and two edges $(v^1, v^2), (v^4, v^3)$. For each undirected edge

$e = (w, u)$ in G the graph G' contains a vertex e^1 and four directed edges $(e^1, w^1), (e^1, u^1), (w^3, e^1), (u^3, e^1)$. Additionally, V' includes a vertex j and for every vertex $v \in V$ the edge set E' includes two directed edges $(v^2, j), (j, v^4)$. Figure 4.4 illustrates an example of this construction.

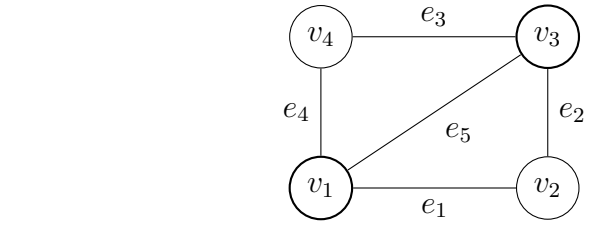
The graph G' has exactly $|E|+4|V|+1$ vertices and $|E'| = 4|E|+4|V|$ edges. Thus, we can build G' from G in polynomial time.

For each undirected edge $e = (u, w)$, the vertices that correspond to e, u, w lie in the same SCC of G' because $(e^1, u^1, u^2, j, u^4, u^3, e^1)$ and $(e^1, w^1, w^2, j, w^4, w^3, e^1)$ are cycles. Consequently, G' is strongly connected. Let $v^1, v^2, v^3, v^4 \in V'$ be the vertices that correspond to vertex $v \in V$ and let $x \in \{v^1, v^2, v^3, v^4\}$. Observe that $G' \setminus \{x\}$ contains a vertex of indegree or outdegree 0. Therefore, all the vertices of G' which correspond to the vertices of G are SAPs.

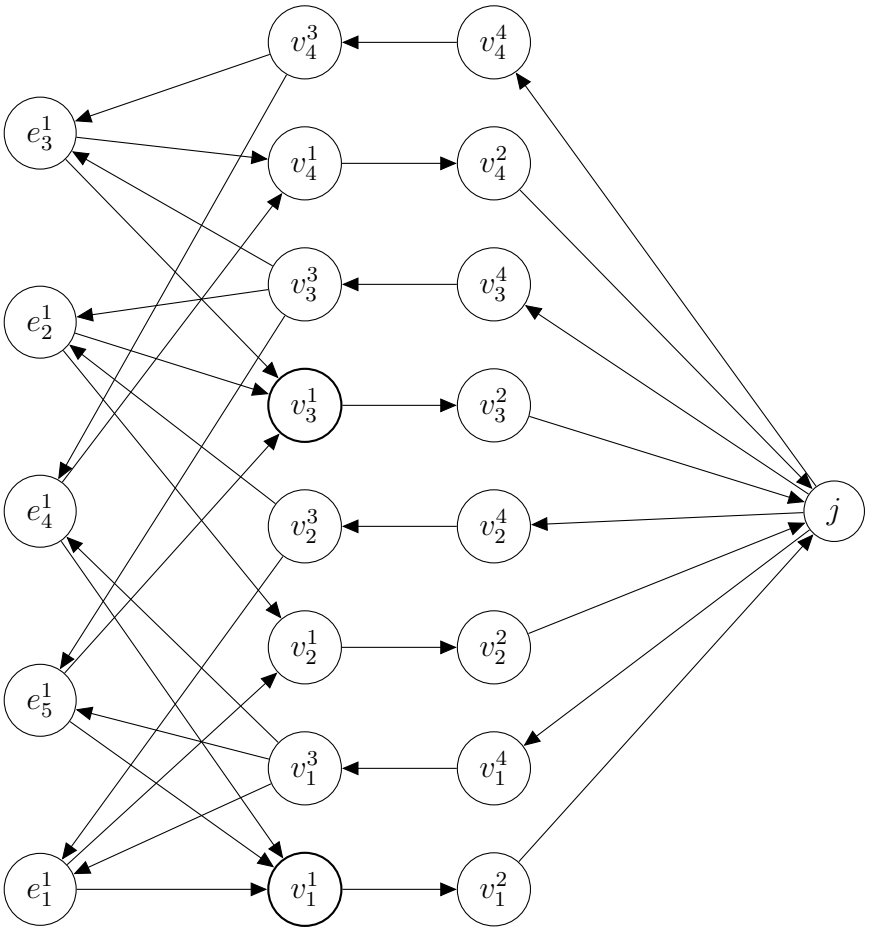
We now identify the 2-directed blocks of G'. For each vertex e^1 of G' which corresponds to undirected edge $e = (u, w) \in E$, there exist two vertex-disjoint paths (e^1, u^1, u^2, j) and (e^1, w^1, w^2, j) from e^1 to j and there exist two vertex-disjoint paths (j, u^4, u^3, e^1) and (j, w^4, w^3, e^1) from j to e^1 in G'. Therefore, the vertices e^1, j are in the same 2-directed block of G'. Let y_1^1, y_2^1 be two vertices in G' which correspond to two distinct edges $y_1, y_2 \in E$, respectively. Note that the vertices y_1^1, y_2^1 are not in the same SCC of $G' \setminus \{j\}$ because in $G' \setminus \{j\}$ there is no path from y_1^1 to y_2^1. Hence, by Lemma 3.1.1, the vertices y_1^1, y_2^1 lie in distinct 2-directed blocks of G'. Furthermore, the vertices of G' which correspond to the vertices of G have indegree or outdegree 1. Therefore, these vertices do not belong to any 2-directed block of G'. As a consequence, for every vertex $e^1 \in V'$ corresponding to undirected edge $e \in E$, the set $\{j, e^1\}$ is a 2-directed block of G'. Because $V' \setminus (\bigcup_{e \in E}\{e^1, j\}) = \bigcup_{v \in V}\{v^1, v^2, v^3, v^4\}$, all the vertices in $V' \setminus (\bigcup_{e \in E}\{e^1, j\}$ are SAPs in G'. All the vertices in $\bigcup_{v \in V}\{v^1, v^2, v^3, v^4\}$ do not lie in any 2-directed block of G' and the directed graph obtained from G' by deleting the set $\bigcup_{v \in V}\{v^1, v^2, v^3, v^4\}$ does not contain any 2-directed block. Therefore, it suffices to show the following.

Claim 4.3.3
The graph G has a vertex cover of size at most l if and only if there exists a vertex set $V^* \subseteq \bigcup_{v \in V}\{v^1, v^2, v^3, v^4\}$ of size at most $r = l$ such that $G' \setminus V^*$ has no 2-directed blocks.

Proof. "⇒": Suppose that U is a vertex cover of G with $|U| \leq l$. Let $V^* = \{v^1 \mid v^1 \in V' \text{ and } v \in U\}$. It is obvious that $|V^*| \leq l$. Assume for a contradiction that $G' \setminus V^*$ has a 2-directed block C^{2d}. By construction $C^{2d} = \{e^1, j\}$ for some



(a) $G = (V, E)$, $U = \{v_1, v_3\}$



(b) $G' = (V', E'), V^* = \{v_1^1, v_3^1\}$

Figure 4.4.: Reducing the vertex cover problem to Problem 4.3.1

edge $e = (u, w) \in E$. Therefore, the vertex e^1 has outdegree at least 2 in the graph $G' \setminus V^*$. This implies that the vertices u^1 and w^1 are in the graph $G' \setminus V^*$. Because neither u^1 nor w^1 belongs to the set V^*, we have $\{u, w\} \cap U = \emptyset$, a contradiction.

"$\Leftarrow$": Suppose that V^* is a subset of $\bigcup_{v \in V} \{v^1, v^2, v^3, v^4\}$ of size at most l such that $G' \setminus V^*$ has no 2-directed blocks. Let $U = \{v \mid \{v^1, v^2, v^3, v^4\} \cap V^* \neq \emptyset$ and $v \in V\}$. It is easy to see that $|U| \leq l$. Assume for a contradiction that U is not a vertex cover of G. Then there is an edge $e = (u, w) \in E$ such that neither u nor w is in U. As a consequence, we have $\{u^1, u^2, u^3, u^4, w^1, w^2, w^3, w^4\} \cap V^* = \emptyset$. Because there exist two vertex-disjoint paths $(e^1, u^1, u^2, j), (e^1, w^1, w^2, j)$ from e^1 to j and two vertex-disjoint paths $(j, u^4, u^3, e^1), (j, w^4, w^3, e^1)$ from j to e^1 in $G' \setminus V^*$. Thus, there exists a 2-directed block C^{2d} in the graph $G' \setminus V^*$ such that $e^1, j \in C^{2d}$, a contradiction. □

Now we prove that Problem 4.3.1 is solvable in time polynomial if the input graph contains only one 2-directed block C^{2d} and the cardinality of C^{2d} is 2. Let $G = (V, E)$ be a strongly connected graph such that $x, y \in V$ and $(x, y) \notin E$. A set $F \subseteq V$ is called an $x - y$-vertex-cut in G if its removal leaves no path from x to y.

Lemma 4.3.4
Let $G = (V, E)$ be a strongly connected graph such that G has only one 2-directed block $C^{2d} = \{x, y\}$. Let V^* be an optimal solution for Problem 4.3.1 on G. Then V^* is neither an $x - y$-vertex-cut nor a $y - x$-vertex cut in G.

Proof. Assume without loss of generality that V^* is an $x - y$-vertex-cut in G. Let v be any SAP of V^*. Then the set $V^* \setminus \{v\}$ is not a solution for Problem 4.3.1 on G. (If the set $V^* \setminus \{v\}$ were a solution, the set V^* would be not optimal.) Therefore, there exist two vertex-disjoint paths from x to y in the graph $G \setminus (V^* \setminus \{v\})$. We consider two cases:

1. $(x, y) \notin E$ Then, by Menger's Theorem for vertex-connectivity, the removal of the vertex v from $G \setminus (V^* \setminus \{v\})$ does not destroy all paths from x to y, but this contradicts that V^* is an $x - y$-vertex-cut.

2. $(x, y) \in E$. In this case, this edge forms a path from x to y in $G \setminus V^*$, a contradiction. □

Lemma 4.3.5
Let $G = (V, E)$ be a strongly connected graph in which there is only one 2-directed block $C^{2d} = \{x, y\}$ and the removal of all the SAPs of G that belong to $V \setminus \{x, y\}$ destroys the 2-directed block C^{2s}. Let V^* be an optimal solution for Problem 4.3.1 on G. Then the following holds.

1. There are no two vertex-disjoint paths from x to y or no two vertex-disjoint paths from y to x in $G \setminus V^*$.
2. If $(x, y) \in E$ and there are no two vertex-disjoint paths from x to y in $G \setminus V^*$, then every minimum $x - y$-vertex-cut in $G \setminus \{(x, y)\}$ such that C_{xy} contains only SAPs is an optimal solution.
3. If $(x, y) \notin E$ and there are no two vertex-disjoint paths from x to y in $G \setminus V^*$, then there is a vertex $v \in V \setminus (V^* \cup \{x, y\})$ such that every minimum $x - y$-vertex-cut of $G \setminus \{v\}$ which contains only SAPs is an optimal solution.

Proof. Since removing the SAPs of G that lie in $V \setminus \{x, y\}$ destroys C^{2s}, we have $V^* \neq \emptyset$.

1. This follows from the fact that the set V^* is a solution for Problem 4.3.1 on G.

2. $(x, y) \in E$ and there do not exist two vertex-disjoint paths from x to y in $G \setminus V^*$. In this case deleting the edge (x, y) from $G \setminus V^*$ destroys all paths from x to y. (If there were a path from x to y in the graph obtained from $G \setminus V^*$ after removing (x, y), there would be two vertex-disjoint paths from x to y in $G \setminus V^*$.) Therefore, V^* is a $x - y$-vertex-cut in $G \setminus \{(x, y)\}$. Let C_{xy} be a minimum $x - y$-vertex-cut in $G \setminus \{(x, y)\}$ such that C_{xy} contains only SAPs. Since $|C_{xy}| \leq |V^*|$ and C_{xy} is a feasible solution for Problem 4.3.1 on G, we have $|V^*| = |C_{xy}|$.

3. $(x, y) \notin E$ and there do not exist two vertex-disjoint paths from x to y in $G \setminus V^*$. By Lemma 4.3.4, there is a path from x to y in $G \setminus V^*$. Because there do not exist two vertex-disjoint paths from x to y in $G \setminus V^*$, by Menger's Theorem for vertex connectivity there is a vertex $v \in V \setminus (V^* \cup \{x, y\})$ such that v lies on each path from x to y in $G \setminus V^*$. Hence, the

set V^* is a $x-y$-vertex-cut in $G \setminus \{v\}$. Let C_{xy} be any minimum $x-y$-vertex-cut in $G \setminus \{v\}$ such that each vertex in this cut is a SAP of G. Then $|C_{xy}| \leq |V^*|$. Because C^{2s} is not a 2-directed block of $G \setminus C_{xy}$, we have $|C_{xy}| = |V^*|$. □

Algorithm 4.3.6

Input: A strongly connected graph $G = (V, E)$ containing only 2-directed block $C^{2d} = \{x, y\}$.
Output: An optimal solution V^* for Problem 4.3.1 on G.

```
1   Compute V^sap.
2   if C^2d is a 2-directed block of G \ (V^sb \ {x,y}) then
3     V* ← ∅.
4   else
5     V* ← V^sap \ {x,y}.
6     if (x,y) ∈ E then
7       find a minimum x − y-vertex cut C_xy of G \ {(x,y)} such that
8       C_xy ⊆ V^sap \ {x,y}.
9       if |C_xy| < |V*| then
10        V* ← C_xy.
11    else
12      for each vertex v ∈ V \ {x,y} do
13        find a minimum x − y-vertex cut C_xy of G \ {v} such that
14        C_xy ⊆ V^sap \ {x,y}.
15        if |C_xy| < |V*| then
16          V* ← C_xy.
17    if (y,x) ∈ E then
18      find a minimum y − x-vertex cut C_yx of G \ {(y,x)} such that
19      C_yx ⊆ V^sap \ {x,y}.
20      if |C_yx| < |V*| then
21        V* ← C_yx.
22    else
23      for each vertex v ∈ V \ {x,y} do
24        find a minimum y − x-vertex cut C_yx of G \ {v} such that
25        C_yx ⊆ V^sap \ {x,y}.
26        if |C_yx| < |V*| then
```

27 $V^* \leftarrow C_{yx}$.
28 **Output** V^*.

The following Lemma shows that the output of Algorithm4.3.6 is an optimal solution.

Lemma 4.3.7
Algorithm 4.3.6 returns an optimal solution V^* for Problem 4.3.1 on a strongly connected graph $G = (V, E)$ when G contains only one 2-directed block $C^{2d} = \{x, y\}$.

Proof. Obviously, when removing the set $V^{sb} \setminus \{x, y\}$ from G does not destroy C^{2d}, then deleting any subset of $V^{sb} \setminus \{x, y\}$ from G leaves x, y in the same 2-directed block. Therefore, in this case $V^* = \emptyset$. Now we consider the case when there is no 2-directed block in $G \setminus (V^{sap} \setminus \{x, y\})$, i.e., $V^* \neq \emptyset$. First we prove that the output V^* of Algorithm 4.3.6 is a feasible solution for Problem 4.3.1 on G. It suffices to consider the following two cases.

1. The output V^* is a minimum $x - y$-vertex cut C_{xy} of $G \setminus \{(x, y)\}$ with $C_{xy} \subseteq V^{sap} \setminus \{x, y\}$. Then there are no two vertex-disjoint paths from x to y in $G \setminus V^*$.

2. The output V^* is a minimum $x - y$-vertex cut C_{xy} of $G \setminus \{v\}$ for some vertex $v \in V \setminus \{x, y\}$ such that $C_{xy} \subseteq V^{sap} \setminus \{x, y\}$. Since $(x, y) \notin E$, the vertex v lies on each path from x to y in $G \setminus V^*$. By Menger's Theorem for vertex connectivity there do not exist two vertex-disjoint paths from x to y in $G \setminus V^*$.

The output V^* is a feasible solution. By Lemma 4.3.5 (2. and 3.), V^* is optimal. □

It remains to explain how to compute minimum vertex cuts that contain only SAPs. As is well known, the problem of computing a minimum capacity $x - y$-vertex-cut of G can be reduced to the problem of finding a minimum edge cut as follows. We construct a directed graph, denoted by $G^{vc} = (V^{vc}, E^{vc})$, from G in the following way: For each vertex $v \in V \setminus \{x, y\}$, we add two vertices v^1, v^2 to V^{vc} and a directed edge (v^1, v^2) to E^{vc}. This edge has the same capacity as $v \in V$. The set V^{vc} contains two vertices x', y' which correspond to $x, y \in V$, respectively. Furthermore, $E^{vc} = \{(v^2, w^1) \mid (v, w) \in E$ and $v, w \notin \{x, y\}\} \cup \{(x', v^1) \mid (x, v) \in E\} \cup \{(v^2, x') \mid (v, x) \in E\} \cup \{(v^2, y') \mid$

$(v, y) \in E\} \cup \{(y', v^1) \mid (y, v) \in E\}$. All edges of E^{vc} corresponding to the edges of E have capacity ∞. Let C'_{xy} be a minimum $x' - y'$-edge cut in G'. Then the set of vertices that correspond to the edges of C'_{xy} is a minimum capacity $x - y$-vertex-cut of G.

Since here we are only interested in calculating minimum vertex-cuts that contain only SAPs, we define a capacity function $w : E^{vc} \to \mathbb{N}$ as follows:

$$w(e') = \begin{cases} 1 & \text{,if } e' \text{ corresponds to } v \in V^{sap} \\ m & \text{,if } e' \text{ corresponds to } v \in V \setminus V^{sap} \\ nm & \text{,if } e' \text{ corresponds to } e \in E \end{cases}$$

When $(x, y) \notin E$, each minimum $x - y$-vertex cut has capacity at most $(n-2)m$. Since each edge of E^{vc} corresponding to an edge of E has capacity of nm, each minimum $x' - y'$-edge cut contains only edges that correspond to vertices of G. Moreover, each minimum $x' - y'$-edge cut $C_{x'y'}$ with capacity $|C_{x'y'}|$ correspond to a minimum $x - y$-vertex cut that contains only SAPs.

Theorem 4.3.8
Algorithm 4.3.6 runs in $O(mn^2)$ time.

Proof. The SAPs of a strongly connected graph can be computed in linear time using the algorithm of Italiano et al. [ILS12]. Lines 2–3 take $O(m)$ time since the 2-directed blocks of a directed graph can be calculated in linear time using the algorithm of Georgiadis et al. [Geo+15b]. A minimum vertex cut can be found in $O(nm)$ time [Orl13; GT14]. Since lines 12–16 and lines 23–27 require $O(mn^2)$ time, the total running time is $O(mn^2)$. □

Figure 4.5 and Figure 4.6 show examples in which Algorithm 4.3.1 returns an optimal solution.

4.4. Destroying a subset of a 2-edge block

In this section we are interested in removing strong bridges that affects only certain vertices in a 2-edge block of a strongly connected graph. We define the problem that we consider in this section as follows.

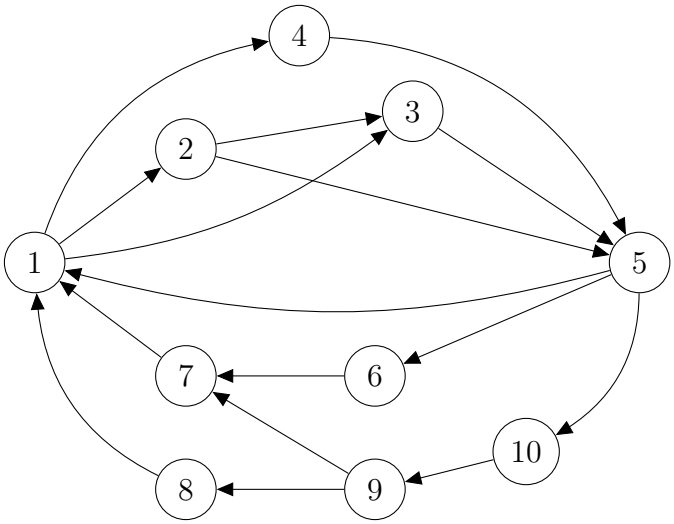


Figure 4.5.: A strongly connected graph $G = (V, E)$ with the SAPs $10, 7, 9, 1, 5$. This graph has only one 2-directed block $C^{2d} = \{1, 5\}$. Algorithm 4.3.6 returns an optimal solution $V^* = \{7, 9\}$, which is a minimum $5 - 1$-vertex cut of $G \setminus \{(5, 1)\}$.

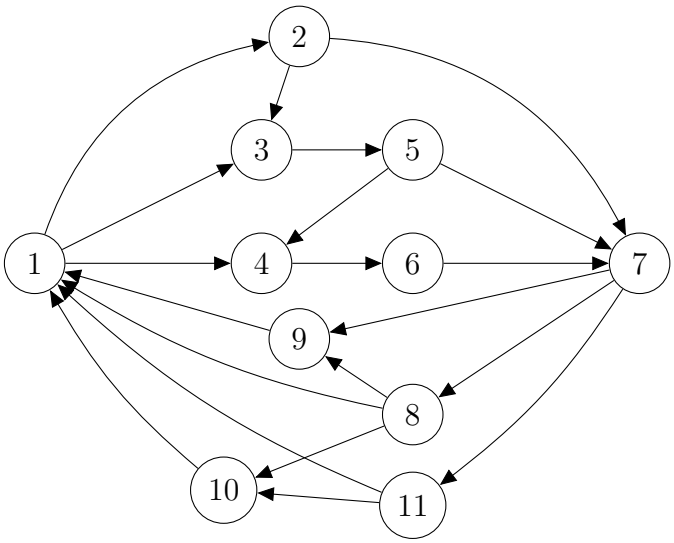


Figure 4.6.: A strongly connected graph $G = (V, E)$ with one 2-directed block $C^{2d} = \{1, 7\}$. Its SAPs are $5, 3, 4, 6, 1, 7$. Algorithm 4.3.6 returns an optimal solution $V^* = \{3, 4\}$, which is a minimum $1 - 7$-vertex cut of $G \setminus \{2\}$.

Problem 4.4.1
Input: A strongly connected graph $G = (V, E)$ and a subset A of a 2-edge block of G such that none of the vertices of A lies in any 2-edge block of the directed graph $G \setminus E^{sb}$.
Task: Find a subset $E^* \subseteq E^{sb}$ of minimum cardinality such that none of the vertices of A belongs to any 2-edge block of $G \setminus E^*$.

We define the decision version of Problem 4.4.1 in the following way: Given a strongly connected graph $G = (V, E)$, a subset A of a 2-edge block of G such that the 2-edge blocks of $G \setminus E^{sb}$ do not contain any vertex of A, and an integer r. Does G have a subset $E^* \subseteq E^{sb}$ with $|E^*| \leq r$ such that none of the vertices of A lies in any 2-edge block of $G \setminus E^*$? We show that this Problem is NP-complete by reducing the vertex cover problem to it.

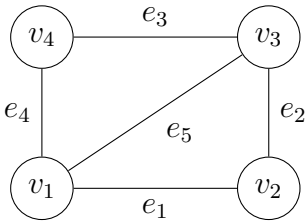
Lemma 4.4.2
The decision version of Problem 4.4.1 is NP-complete.

Proof. It is clear that Problem 4.4.1 is in NP. Given an instance $(G = (V, E), l)$ of the vertex cover problem, we build an instance $(G' = (V', E'), r)$ of Problem 4.4.1 as follows. For every vertex $v \in V$ the graph G' contains four vertices v^1, v^2, v^3, v^4 and two directed edges $(v^1, v^2), (v^4, v^3)$. For every edge $e = (x, y) \in E$, the graph G' contains a vertex e^1 and four directed edges (e^1,x^1),(x^3, e^1),(e^1,y^1),(y^3,e^1). Furthermore, we add three vertices w_1,w_2,w_3 to V' and six directed edges (w_1,w_2),(w_2, w_1),(w_2,w_3),(w_3,w_2),(w_1,w_3),(w_3, w_1) to E'. For every vertex $v \in V$ the set E' contains directed edges $(v^2, w_i), (w_i, v^4)$ for $i \in \{1, 2, 3\}$. An example of this construction is illustrated in Figure 4.7.

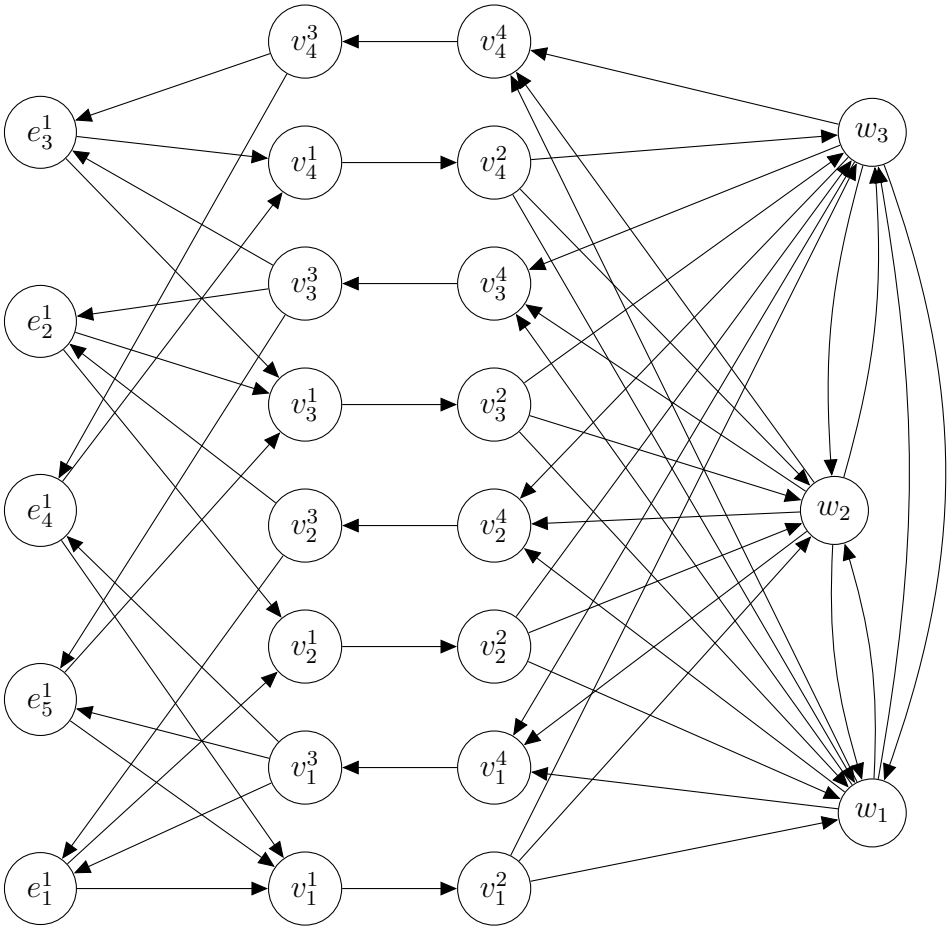
We can build G' in polynomial time because the directed graph G' has $|E| + 4|V| + 3$ vertices and $4|E| + 8|V| + 6$ edges.

Note that the set of vertices $\{w_1, w_2, w_3\}$ is a subset of a SCC of G'. Let $w_i \in \{w_1, w_2, w_3\}$. For all edges e incident to vertex x in G, the vertices of G' that correspond to e, x lie in the same SCC of G' since $(e^1, x^1, x^2, w_i, x^4, x^3, e^1)$ forms a cycle in G'. Consequently, G' is strongly connected.

We now show that the strongly connected graph $G' = (V', E')$ has only one 2-edge block $C^{2e} \subseteq V'$ that includes the vertices w_1, w_2, w_3 and the vertices that correspond to the edges of G. All the vertices of G' that correspond to the vertices of V have indegree or outdegree 1. Therefore, none of them lies in any 2-edge block of G'. For each vertex $e^1 \in E'$ which correspond to undirected edge $e \in E$, the graph G' contains two edge-disjoint paths from e^1 to w_1 and



(a) $G = (V, E)$, $U = \{v_2, v_3, v_4\}$



(b) $G' = (V', E'), A = \{e_1^1, e_2^1, e_3^1, e_4^1, e_5^1\}, E^* = \{(v_2^1, v_2^2), (v_3^1, v_3^2), (v_4^1, v_4^2)\}$

Figure 4.7.: Reducing the vertex cover problem to Problem 4.4.1

two edge-disjoint paths from w_1 to e^1. Hence, the vertices e^1 and w_1 are in the same 2-edge block C^{2e} of G'. Since the vertices w_1, w_2, w_3 are in the same 2-edge block of G', by Lemma 3.3.2 we have $\{w_1, w_2, w_3\} \subseteq C^{2e}$.

Let A be the set of the vertices in G' that correspond to the edges of G. The set of edges $\{(v^1, v^2), (v^4, v^3) \mid v \in V\}$ are strong bridges since removing any edge of this set from G' increases the number of the SCCs of G'. Because none of the vertices in A lies in any 2-edge block of $G' \setminus (\bigcup_{v \in V}\{(v^1, v^2), (v^4, v^3)\})$, the 2-edge blocks of $G' \setminus E^{sb}$ do not contain any vertex of A.

Claim 4.4.3
There is a vertex cover U in G such that $|U| \leq l$ if and only if G' has a subset $E^* \subseteq E^{sb}$ with $|E^*| \leq r = l$ such that none of the vertices of A lies in any 2-edge block of $G' \setminus E^*$.

Proof. "⇒": Suppose that G has a vertex cover U with $|U| \leq l$. Let $E^* = \{(v^1, v^2) \mid (v^1, v^2) \in E' \text{ and } v \in U\}$. Clearly, $|E^*| \leq l$. All the edges of E^* are strong bridges in G'. Let e^1 be a vertex in A such that e^1 corresponds to undirected edge $e = (u, w) \in E$. Let x be a vertex in $C^{2e} \setminus \{e^1\}$. Since U is a vertex cover of G, we have $\{u, w\} \cap U \neq \emptyset$. Assume without loss of generality that $u \in U$. Then $(u^1, u^2) \in E^*$. The vertex u^1 has outdegree 0 and the vertex w^1 has outdegree at most 1 in the directed graph $G' \setminus E^*$. We consider the following cases.

1. The outdegree of w^1 is 1. Then each path from e^1 to x contains the strong bridge (w^1, w^2). Thus, by Menger's Theorem for edge connectivity there are no two edge-disjoint paths from e^1 to x in $G' \setminus E^*$.

2. The outdegree of w^1 is 0. In this case there is no path from e^1 to x in $G' \setminus E^*$.

"⇐": Suppose that G' has a subset $E^* \subseteq E^{sb}$ with $|E^*| \leq l$ such that none of the 2-edge blocks of the directed graph $G' \setminus E^*$ contains any vertex of A. Let $U = \{v \mid \{(e^1, v^1), (v^1, v^2), (v^2, w_1), (v^2, w_2), (v^2, w_3), (w_3, v^4), (w_2, v^4), (w_1, v^4), (v^4, v^3), (v^3, e^1)\} \cap E^* \neq \emptyset \text{ and } v \in V\}$. Suppose by contradiction that U is not a vertex cover of G. Then there are at least two vertices x, y in G such that $e = (x, y) \in E$ and $\{x, y\} \cap U = \emptyset$. This implies that for each vertex $w_i \in \{w_1, w_2, w_3\}$ there exist two edge-disjoint from e^1 to w_i and two edge-disjoint paths from w_i to e^1 in $G' \setminus E^*$. Hence, the vertex e^1 belongs to a 2-edge block of $G' \setminus E^*$, a contradiction. □

Lemma 4.4.4
Problem 4.4.1 remains NP-complete when A contains only one vertex.

Proof. In case the cardinality of A is 1, we modify the proof of Lemma 4.4.2 as follows. We remove the vertices w_2, w_3 together with their incident edges from G'. We call the remaining graph again G'. This graph is strongly connected and has two edge-disjoint paths from e^1 to w_1 and two edge-disjoint paths from w_1 to e^1 for every vertex e^1 which correspond to edge $e \in E$. Thus, the vertex w_1 and the vertices of G' that correspond to the edges of G form a 2-edge block of G'. We choose $A = \{w_1\}$. Similar argument as in Claim 4.4.3 shows that the graph G has a vertex cover U of size at most l if and only if there exists a set $E^* \subseteq E^{sb}$ of cardinality at most $r = l$ such that the vertex $w_1 \in A$ does not belong to any 2-edge block of $G' \setminus E^*$. □

Next we show that there is an $(r-1)$ approximation algorithm for Problem 4.4.1 when the set A contains only one vertex w, where r is the cardinality of the 2-edge block C^{2e} of G which contains the set A.

We define a capacity function $w : E \to \mathbb{N}$ as follows:

$$w(e) = \begin{cases} 1 & ,\text{if } e \in E^{sb} \\ 2m & ,\text{if } e \in E \setminus E^{sb} \end{cases}$$

Algorithm 4.4.5

Input: A strongly connected graph $G = (V, E)$, a set $A \subseteq C^{2e}$ with $A = \{w\}$ such that w does not belong to any 2-edge block of $G \setminus E^{sb}$, where C^{2e} is a 2-edge block of G.

Output $E_w \subseteq E^{sb}$ such that w does not lie in any 2-edge block of $G \setminus E_w$.

```
1   E_w ← ∅.
2   for each vertex u ∈ C^{2e} \ {w} do
3     E_u ← E^{sb}.
4     for each edge e ∈ E do
5       find a minimum w − u-cut C_wu and a minimum u − w-cut C_uw
6       in G \ {e}.
7       if |C_wu| < |E_u| then
8         E_u ← C_wu.
9       if |C_uw| < |E_u| then
```

10 $E_u \leftarrow C_{uw}$.
11 $E_w \leftarrow E_w \cup E_u$.
12 **Output** E_w.

Lemma 4.4.6
Let E^* be an optimal solution. Then $|E^*| \geq \max\{|E_u| \mid u \in C^{2e} \setminus \{w\}$ and E_u is the set computed in lines 3–10 of Algorithm 4.4.5$\}$.

Proof. Let u be a vertex in $C^{2e} \setminus \{w\}$. There are three cases to consider:

1. There is no path from w to u in $G \setminus E^*$. In this case the set E^* is a $w-u$-cut of G. Let e be an edge in G. We consider two cases.
   a) $e \notin E^*$. Then E^* is a $w-u$-cut in $G \setminus \{e\}$. Since the capacity of E^* is $|E^*|$, then for each minimum $w-u$-cut C_{wu} in $G \setminus \{e\}$, the capacity of C_{wu} is not greater than $|E^*|$.
   b) $e \in E^*$. Then $E^* \setminus \{e\}$ is a $w-u$-cut in $G \setminus \{e\}$. Consequently, for every minimum $w-u$-cut C_{wu} in $G \setminus \{e\}$, the capacity of C_{wu} is less than or equal to $|E^*| - 1$.
2. There does not exist a path from u to w in $G \setminus E^*$. Similar to the proof of case 1.
3. There exist paths from w to u and from u to w in $G \setminus E^*$. We assume without loss of generality that there do not exist two edge-disjoint paths from w to u in $G \setminus E^*$. By Menger's Theorem for edge connectivity there is an edge $e \in E \setminus E^*$ such that the graph $G \setminus (E^* \cup \{e\})$ does not contain any path from w to u. Therefore, the set E^* is a $w-u$-cut of $G \setminus \{e\}$. For every minimum $w-u$-cut C_{wu} in $G \setminus \{e\}$ the capacity of C_{wu} is not greater that $|E^*|$.

Since $E^* \subseteq E^{sb}$, we have $|E^*| \leq m$. Because each edge in $E \setminus E^{sb}$ has capacity $2m$ and the capacity of E_u is less than or equal to $|E^*|$, the capacity of E_u is $|E_u|$. □

The following Lemma shows that the output of Algorithm 4.4.5 is a feasible solution for Problem 4.4.1 when $|A| = 1$.

Lemma 4.4.7
Given a strongly connected graph $G = (V, E)$ and a set $A = \{w\}$ such that A is a subset of a 2-edge block in G and w does not belong to any 2-edge block of $G \setminus E^{sb}$. Let E_w be the output of Algorithm 4.4.5. Then $E_w \subseteq E^{sb}$ and w does not lie in any 2-edge block of $G \setminus E_w$.

Proof. Let u be any vertex in $C^{2e} \setminus \{w\}$. Let E_u be the set of edges computed in lines 3–10 of Algorithm 4.4.5. By Lemma 4.4.6 we have $E_u \subseteq E^{sb}$. Since $E_w = \bigcup_{u \in C^{2e} \setminus \{w\}} E_u$, we have $E_w \subseteq E^{sb}$. Now we prove that w, u are not in the same 2-edge block of $G \setminus E_u$. We consider three cases:

1. $E_u = E^{sb}$. Obviously, w, u are not in the same 2-edge block of $G \setminus E_u$.
2. There is an edge $e \in E$ such that E_u is a minimum $w - u$-cut in $G \setminus \{e\}$. Then $e \in E \setminus E_u$. Assume for a contradiction that the vertices w, u are in the same 2-edge block of $G \setminus E_u$. Then there exist two edge-disjoint paths from w to u and two edge-disjoint paths from u to w in $G \setminus E_u$. Thus, by Menger's Theorem for edge connectivity, for each edge $e \in E \setminus E_u$ there is a path from w to u in $G \setminus (\{e\} \cup E_u)$. It follows that E_u is not a $w - u$-cut in $G \setminus \{e\}$, a contradiction.
3. There exist an edge $e \in E$ such that E_u is a minimum $u - w$-cut in $G \setminus \{e\}$. The proof in this case is similar to Case 2.

For every vertex $u \in C^{2e} \setminus \{w\}$ the vertices w, u are not in the same 2-edge block of $G \setminus E_w$ since $E_w = \bigcup_{u \in C^{2e} \setminus \{w\}} E_u$. Because the 2-edge blocks are disjoint, the vertex w does not lie in any 2-edge block of $G \setminus E_w$. □

Lemma 4.4.8
Let E^* be an optimal solution and let E_w be the output of Algorithm 4.4.5. Then $|E_w| \leq (r-1)|E^*|$, where r is the cardinality of the 2-edge block C^{2e} that contains $A = \{w\}$.

Proof. By Lemma 4.4.6 we have $|E^*| \geq \max\{|E_u| \mid u \in C^{2e} \setminus \{w\}$ and E_u is the set computed in lines 3–10 of Algorithm 4.4.5 $\}$. Since $E_w = \bigcup_{u \in C^{2e} \setminus \{w\}} E_u$, we have $|E_w| \leq \sum_{u \in C^{2e} \setminus \{w\}} |E_u| \leq (|C^{2e}| - 1) \cdot \max\{|E_u| \mid u \in C^{2e} \setminus \{w\}\} \leq (r-1)|E^*|$. □

Theorem 4.4.9
Algorithm 4.4.5 runs in $O(n^2m^2)$ time.

An example in Figure 4.8 shows that the approximation ratio of Algorithm 4.4.5 is tight.

Consider the strongly connected graph G in Figure 4.8 which has one 2-edge block $C^{2e} = \{1, 2, 3, 4\}$. Let $A = \{1\}$. Assume that Algorithm 4.4.5 considers the vertices of $C^{2e} \setminus A$ in the following order $2, 3, 4$. Assume that $E_2 = \{(6, 8), (1, 5)\}$ is the set that is computed by the algorithm in the first iteration of for loop (in lines 3–10). Although E_2 is a feasible solution for Problem 4.4.1 on G, the algorithm considers the vertices 3,4 in order to compute E_3, E_4 and returns $E_1 = E_2 \cup E_3 \cup E_4$. In order to avoid computing unnecessary sets (like E_3 and E_4), check whether E_w is a feasible solution after executing line 10 in each iteration of for loop. If it is, stop and return E_w. Of course, this modification preserves the approximation guarantee of Algorithm 4.4.5.

4.5. Changing 2-strong blocks by removing SAPs

In this section we discuss the following problem.

Problem 4.5.1
Input: A strongly connected graph $G = (V, E)$ with the 2-strong blocks $C_1^{2s}, C_2^{2s}, \ldots, C_t^{2s}$.
Task: Find a minimum cardinality subset $V^* \subseteq V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2s})$ such that the directed graphs $G \setminus V^*$ and $G \setminus (V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2s}))$ have the same 2-strong blocks.

Let $G = (V, E)$ be a strongly connected graph. Problem 4.5.1 is NP-hard for G when the 2-strong blocks of G coincide with the 2-directed blocks. Here we show that Problem 4.5.1 remains NP-hard when each 2-strong block of G has cardinality 2 and is not a 2-directed block.

The decision version of Problem 4.5.1 can be formulated as follows. Given a strongly connected graph $G = (V, E)$ and an integer r. Does there exist a subset $V^* \subseteq V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2s})$ such that V^* has at most r vertices and the 2-strong blocks of $G \setminus V^*$ coincide with the 2-strong blocks of $G \setminus (V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2s}))$, where $C_1^{2s}, C_2^{2s}, \ldots, C_t^{2s}$ are the 2-strong blocks of G?

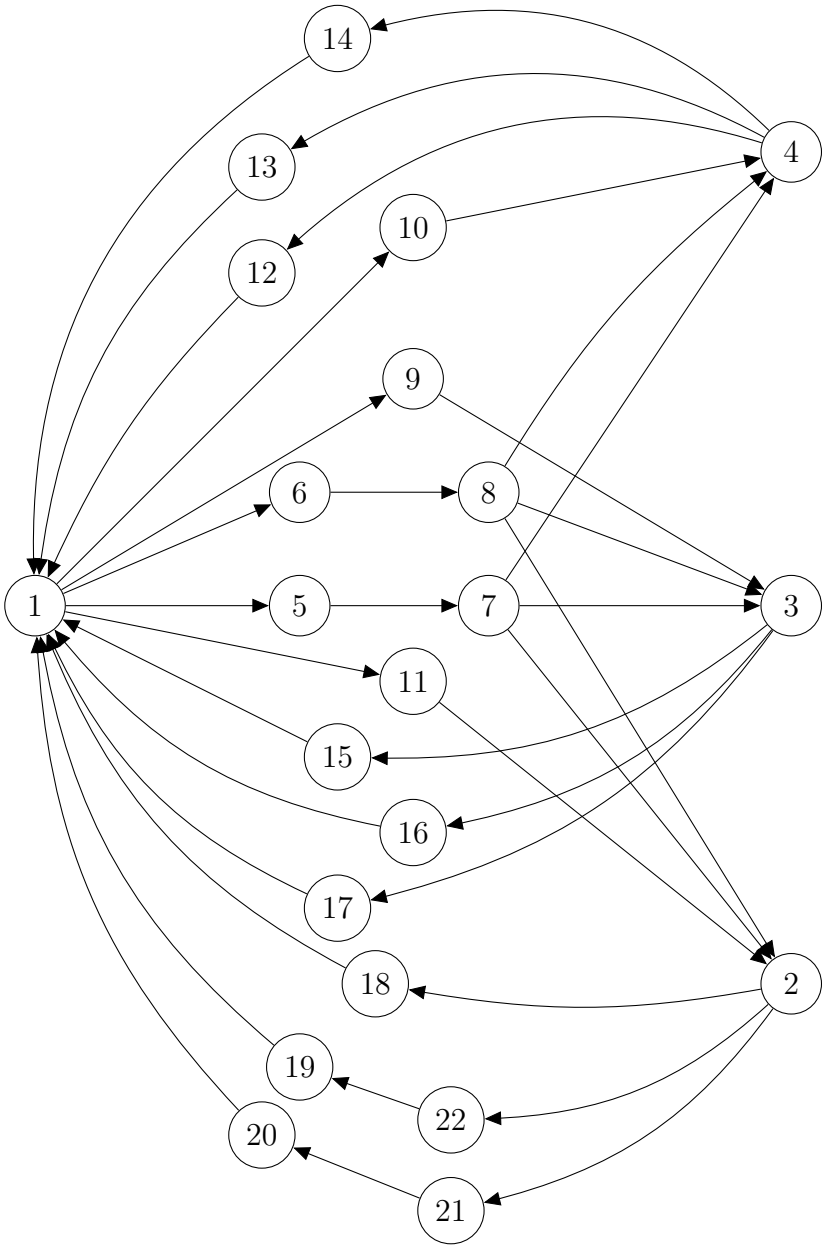


Figure 4.8.: A strongly connected graph $G = (V, E)$ which contains only one 2-edge block $C^{2e} = \{1, 2, 3, 4\}$. For $A = \{1\}$, the set $E^* = \{(6, 8),(5, 7)\}$ is an optimal solution for Problem 4.4.1 on G. A possible output of Algorithm 4.4.5 is $E_1 = \{(12, 1), (13, 1), (3, 15), (3, 16), (21, 20), (22, 19)\}$. Notice that $|E_1| = (|C^{2e}| - 1)|E^*|$.

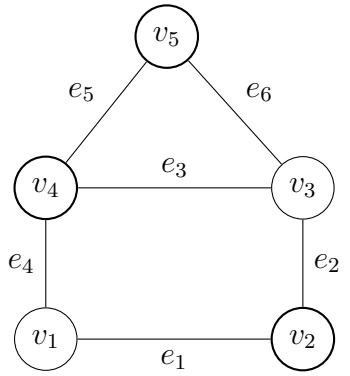
Lemma 4.5.2
The decision version of Problem 4.5.1 is NP-complete.

Proof. It is obvious that Problem 4.5.1 belongs to NP. Suppose that $(G = (V, E), l)$ is an instance of the vertex cover problem. We construct an instance $(G' = (V', E'), r)$ for Problem 4.5.1 in the following way: For every vertex $v \in V$ we place three vertices v^1, v^2, v^3 into V' and two directed edge $(v^1, v^2), (v^2, v^3)$ into E'. Moreover, for each edge $e = (u, w) \in E$ we add two new vertices e^1, e^2 to V' and five directed edges $(e^2, e^1), (e^1, u^1), (e^1, w^1), (u^3, e^2), (w^3, e^2)$ to E'. The directed graph G' can be constructed from G in time polynomial since $|V'| = 3|V| + 2|E|$ and $|E'| = 2|V| + 5|E|$. Figure 4.9 gives an example for this construction.

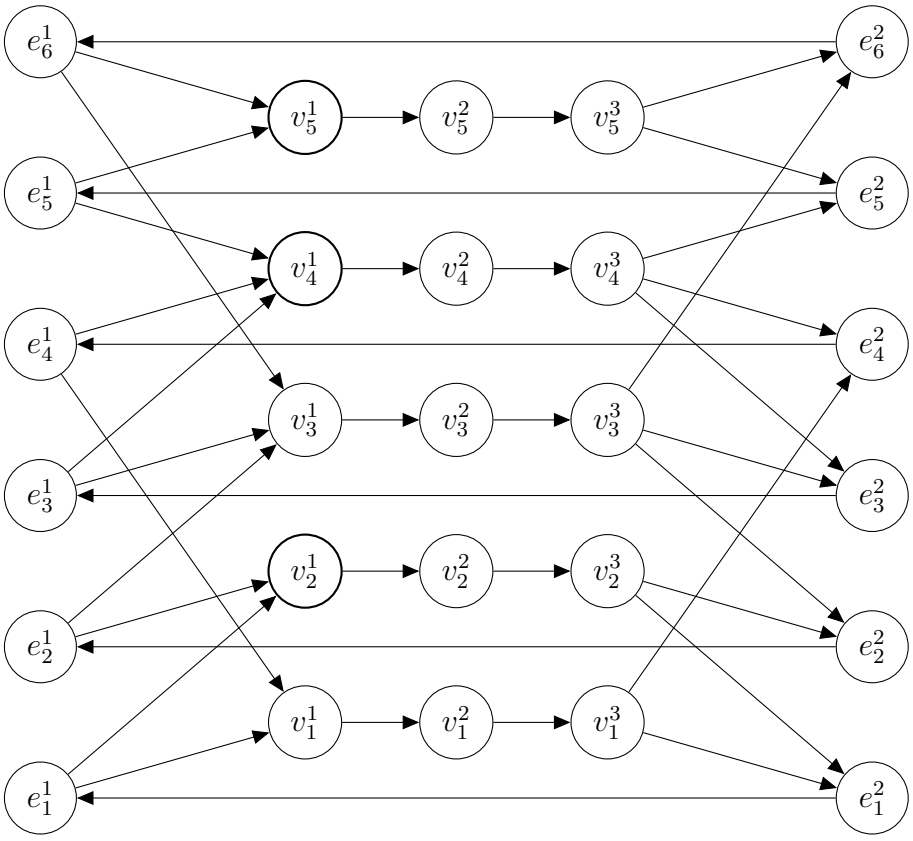
Let $e = (x, y)$ be an edge in G. Consider the vertices of G' that correspond to the edge e and the vertice x, y. All these vertices are in the same SCC of G' since $(e^1, x^1, x^2, x^3, e^2, e^1)$ and $(e^1, y^1, y^2, y^3, e^2, e^1)$ form cycles in G' and both cycles have vertices in common. This implies that G' is strongly connected. Note that each vertex in the strongly connected graph G' has indegree or outdegree 1. Thus, G' contains no 2-directed blocks.

Now we identify the 2-strong blocks of G'. Let v be a vertex in G and let v^1, v^2, v^3 be the vertices in G' which correspond to the vertex v. Since there is no path from v^2 to v^1 in $G' \setminus \{v^3\}$, then the vertices v^1, v^2 do not lie in the same 2-strong block of G'. Moreover, for every vertex $x \in V' \setminus \{v^1, v^2\}$ the vertices v^1, x do not lie in the same SCC of $G' \setminus \{v^2\}$ because v^1 has outdegree 0 in $G' \setminus \{v^2\}$. As a consequence, v^1 does not belong to any 2-strong blocks of G'. For any vertex $y \in V' \setminus \{v^1, v^2\}$, the vertices y, v^2 are not in the same SCC of $G' \setminus \{v^1\}$. Hence, v^2 does not belong to any 2-strong block of G'. Note also that v^3 does not belong to any 2-strong block of G' because for any vertex $x \in V' \setminus \{v^3, v^2, v^1\}$ the graph $G' \setminus \{v^2\}$ does not contain any path from x to v^3. Therefore, all vertices of G' that correspond the vertices of G do not lie in the 2-strong blocks of G'. For each pair of vertices $e^1, e^2 \in V'$ that correspond to undirected edge $e \in E$ the vertices e^1, e^2 lie in the same 2-strong block of G' since there exist two vertex-disjoint paths from e^1 to e^2 and there is an edge (e^2, e^1) in E'. Furthermore, For each vertex $x \in V' \setminus \{e^1, e^2\}$, the vertex x is not reachable from e^2 in $G' \setminus \{e^1\}$ and the vertex e^1 is not reachable from x in $G' \setminus \{e^2\}$. Consequently, $\{e^1, e^2\}$ is a 2-strong block of G'.

Notice that $V' \setminus (\bigcup_{e \in E} \{e^1, e^2\}) = \bigcup_{v \in V} \{v^1, v^2, v^3\}$ and all the vertices of the set $\bigcup_{v \in V} \{v^1, v^2, v^3\}$ are SAPs of G' since removing any of them from G' leaves a directed graph containing a vertex of indegree or outdegree 0. Additionally, the



(a) $G = (V, E)$, $U = \{v_2, v_4, v_5\}$



(b) $G' = (V', E'), V^* = \{v_2^1, v_4^1, v_5^1\}$

Figure 4.9.: Reducing the vertex cover problem to Problem 4.5.1

directed graph obtained from G' after removing the vertices of $\bigcup_{v \in V}\{v^1, v^2, v^3\}$ has no 2-strong blocks. Thus, it suffices to prove the following.

Claim 4.5.3
The graph G has a vertex cover U of cardinality at most l if and only if there exists a subset $V^* \subseteq \bigcup_{v \in V}\{v^1, v^2, v^3\}$ of cardinality $r = l$ such that $G' \setminus V^*$ does not have any 2-strong block.

Proof. "⇒": Suppose that there is a vertex cover U in G with $|U| \leq l$. Let $V^* = \{v^1 \mid v \in U\}$. Since $|V^*| = |U|$, we have $|V^*| \leq l$. Let C^{2s} be any 2-strong block of G'. Then $C^{2s} = \{e^1, e^2\}$ for some undirected edge $e = (u, w) \in E$. Because $U \cap \{u, w\} \neq \emptyset$, we have $\{u^1, w^1\} \cap V^* \neq \emptyset$. There are two cases to consider:

1. $\{u^1, w^1\} \subseteq V^*$. Then the vertex e^1 has outdegree 0 in $G' \setminus V^*$. Therefore, neither e^1 nor e^2 lie in any 2-strong block in $G' \setminus V^*$.

2. Either u^1 or w^1 is in V^*. Assume without loss of generality that $w^1 \in V^*$. Then the vertex e^2 is no longer reachable from e^1 in the directed graph obtained from $G' \setminus V^*$ by deleting the vertex u^1. Hence, the set $\{e^1, e^2\}$ is not a 2-strong block of $G' \setminus V^*$.

"⇐": Suppose that V^* is a subset of $\bigcup_{v \in V}\{v^1, v^2, v^3\}$ with $|V^*| \leq l$ such that $G' \setminus V^*$ does not have any 2-strong block. Let $U = \{v \mid \{v^1, v^2, v^3\} \cap V^* \neq \emptyset$ and $v \in V\}$. Obviously, $|U| \leq l$. Let $e = (u, w)$ be an edge in G. Then set $\{e^1, e^2\}$ is a 2-strong block of G'. This set is also a 2-strong block of the subgraph induced on $\{e^1, e^2, u^1, u^2, u^3, w^1, w^2, w^3\}$ since this subgraph contains two vertex-disjoint paths from e^1 to e^2 and the edge (e^2, e^1). Because $G' \setminus V^*$ does not have any 2-strong blocks, then none of its subgraphs has 2-strong blocks. Since $G' \setminus V^*$ contains the edge (e^2, e^1) and does not contain an edge (e^1, e^2), there are no 2-vertex-disjoint paths from e^1 to e^2 in $G' \setminus V^*$. Therefore, we have $V^* \cap \{u^1, u^2, u^3, w^1, w^2, w^3\} \neq \emptyset$. It follows that $\{u, w\} \cap U \neq \emptyset$. Thus, the set U is a vertex cover of G. □

4.6. Changing some 2-strong blocks by removing non-SAPs

Let S be a subset of the 2-strong blocks of a strongly connected graph $G = (V, E)$. Note that deleting the vertices of $V \setminus V^{sap}$ that do not belong to any

2-strong block in S may change these blocks. In this section we deal with the following problem.

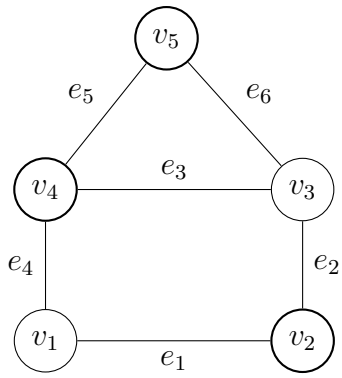
Problem 4.6.1
Input: A strongly connected graph $G = (V, E)$ with the 2-strong blocks $C_1^{2s}, C_2^{2s}, \ldots, C_t^{2s}$, a set $S \subseteq \{C_1^{2s}, C_2^{2s}, \ldots, C_t^{2s}\}$ such that the removal of the vertices that are not SAPs and do not lie in $\bigcup_{C^{2s} \in S} C^{2s}$ destroys all the 2-strong blocks that are in S.
Task: Find a minimum cardinality set $V^* \subseteq V \setminus (\bigcup_{C^{2s} \in S} C^{2s})$ such that none of the elements of S is a strong block of $G \setminus V^*$ and none of the vertices of V^* is a SAP of G.

Let G be a strongly connected graph. Let $C_1^{2s}, C_2^{2s}, \ldots, C_t^{2s}$ be the 2-strong blocks of G. If these blocks and the 2-directed blocks of G are identical and S includes all these blocks, then by Lemma 4.2.2, Problem 4.6.1 is NP-hard on G. In this section we prove that Problem 4.6.1 remains NP-hard when the 2-strong blocks of G are not necessarily identical to the 2-directed blocks of G and S is a subset of $\{C_1^{2s}, C_2^{2s}, \ldots, C_t^{2s}\}$ such that the removal of the vertices that are not SAPs and do not lie in $\bigcup_{C^{2s} \in S} C^{2s}$ destroys all the 2-strong blocks that are in S.

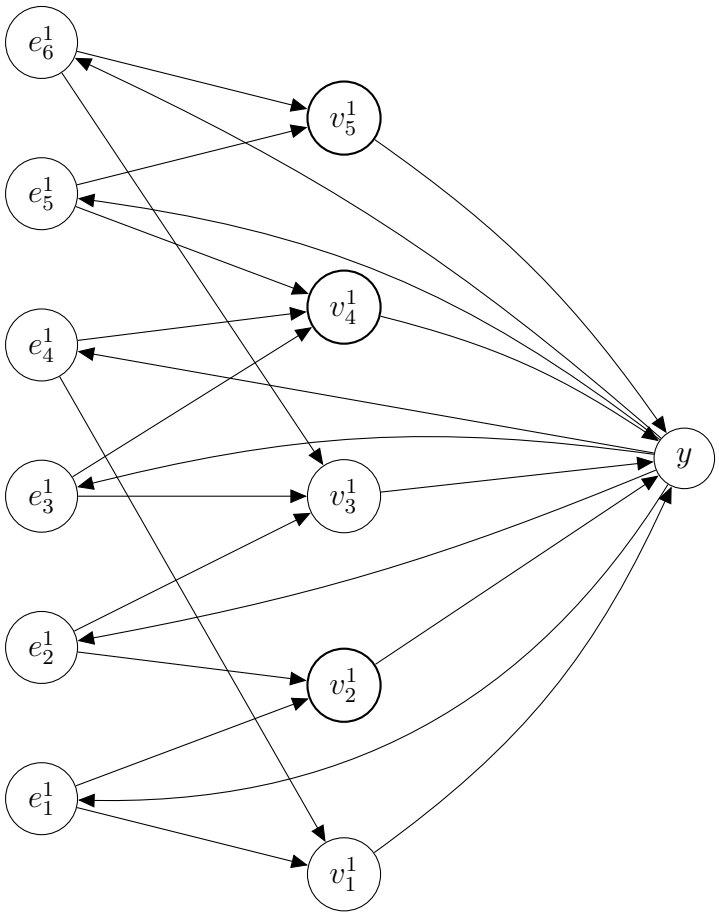
We define the decision version of Problem 4.6.1 as follows. Given a strongly connected graph $G = (V, E)$ with the 2-strong blocks $C_1^{2s}, C_2^{2s}, \ldots, C_t^{2s}$, a set $S \subseteq \{C_1^{2s}, C_2^{2s}, \ldots, C_t^{2s}\}$ such that deleting the vertices which belong to $V \setminus V^{sap}$ and are not in $\bigcup_{C^{2s} \in S} C^{2s}$ destroys all the 2-strong blocks that belong to S, and an integer r. Does there exist a subset $V^* \subseteq (V \setminus V^{sap}) \setminus (\bigcup_{C^{2s} \in S} C^{2s})$ such that none of the elements of S is a strong block of $G \setminus V^*$?

Lemma 4.6.2
The decision version of Problem 4.6.1 is NP-complete

Proof. Problem 4.6.1 is clearly in NP. Let $(G = (V, E), l)$ be an input of the vertex cover problem. An input $(G' = (V', E'), r)$ of Problem 4.6.1 can be built as follows. The set V' contains a vertex v^1 for every vertex $v \in V$ and a vertex e^1 for each edge $e \in E$. We place a new vertex y in V'. Additionally, the set E' includes a directed edge (v^1, y) for every vertex $v \in V$ and three edges (e^1, v^1),(e^1, w^1),(y,e^1) for every edge $e = (v, w) \in E$. Figure 4.10 gives an example for this construction. Since $|V'| = |V| + |E| + 1$ and $|E'| = 3|E| + |V|$, the directed graph G' can be constructed from G in time polynomial.



(a) $G = (V, E)$, $U = \{v_2, v_4, v_5\}$



(b) $G' = (V', E'), V^* = \{v_2^1, v_4^1, v_5^1\}$

Figure 4.10.: Reducing the vertex cover problem to Problem 4.6.1

For every edge e incident to vertex v in G, the graph G' has a cycle (e^1, v^1, y, e^1). Therefore, the graph G' is strongly connected. For every vertex e^1 of G' that corresponds to undirected edge $e = (u, w) \in E$, there are two vertex-disjoint paths from e^1 to y in G', namely (e^1, u^1, y) and (e^1, w^1, y). Because $(y, e^1) \in E'$, the vertices e^1, y are in the same 2-strong block of G'. Moreover, for every vertex $x \in V' \setminus \{e^1, y\}$, the vertices e^1, x do not lie in the same SCC of $G' \setminus \{y\}$. Therefore, e^1, x are not in the same 2-strong block of G'. This implies that $\{e^1, y\}$ is a 2-strong block of G'. Let $S = \{\{e^1, y\} \mid e \in E\}$. Notice that S does not contain all the 2-strong blocks of G. All the vertices of G' that correspond to the vertices of G are not SAPs. Thus, none of the vertices of $V' \setminus (\bigcup_{C^{2s} \in S} C^{2s})$ is a SAP of G'. Note that the removal of these vertices destroys all the 2-strong blocks that belong to S.

Next we show the following.

Claim 4.6.3
The graph G has a vertex cover U with $|U| \leq l$ if and only if there is a subset $V^* \subseteq V' \setminus (\bigcup_{C^{2s} \in S} C^{2s})$ with $|V^*| \leq l$ such that the removal of V^* from G' destroys all the 2-strong blocks that are in S.

Proof. "⇒": Let U be a vertex cover of G such that $|U| \leq l$. Let $V^* = \{v^1 \mid v \in U \text{ and } v^1 \in V'\}$. Since $|V^*| = |U|$, we have $|V^*| \leq l$. We have seen that none of the elements of V^* is a SAP of G'. It remains to prove that none of the elements of S is a 2-stong block in the directed graph $G' \setminus V^*$. Let C^{2s} be an element of S. Then $C^{2s} = \{e^1, y\}$, where e^1 corresponds to undirected edge $e = (u, w) \in E$. Since $\{u, w\} \cap U \neq \emptyset$, we have $\{u^1, w^1\} \cap V^* \neq \emptyset$. There are two cases to consider.

1. Either u^1 or w^1 belongs to V^*. Assume without loss of generality that $u^1 \in V^*$. Since the vertex w^1 lies on all paths from e^1 to y in $G' \setminus V^*$, the vertices e^1, y are in distinct SCCs of $G' \setminus (V^* \cup \{w^1\})$. Hence, C^{2s} is not a 2-strong block in $G' \setminus V^*$.

2. $\{u^1, w^1\} \subseteq V^*$. Because the vertex e^1 has outdegree 0 in $G' \setminus V^*$, there is no path from e^1 to y in $G' \setminus V^*$. Thus, the set C^{2s} is not a 2-strong block of $G' \setminus V^*$.

"⇐": Let V^* a subset of $V' \setminus (\bigcup_{C^{2s} \in S} C^{2s})$ with $|V^*| \leq l$ such that the removal of V^* from G' destroys all the 2-strong blocks that belong to S. It is clear that all the vertices in V^* correspond to vertices of G. Let $U = \{v \mid v^1 \in V^* \text{ and }$

$v \in V\}$. The set U has at most l vertices since $|U| = |V^*|$. Assume for a contradiction that the set U is not a vertex cover of G. Then there is an edge $e = (u, w) \in E$ such that neither u nor w belongs to U. This implies that neither u^1 nor w^1 is in V^*. Therefore, there are two vertex-disjoint paths from e^1 to y in $G' \setminus V^*$, namely (e^1, u^1, y) and (e^1, w^1, y). Since (y, e^1) is an edge in $G' \setminus V^*$, the vertices e^1, y lie in the same 2-strong block of $G' \setminus V^*$. For any vertex $x \in V' \setminus (V^* \cup \{e^1, y\})$, the vertices x, e^1 do not lie in the same 2-strong block of G'. Thus, x, e^1 are not in the same 2-strong block of $G' \setminus V^*$. It follows that $\{e^1, y\}$ is a 2-strong block of $G' \setminus V^*$, but this block is an element of S, a contradiction. □

4.7. Strong bridges that disrupt 2-edge blocks

Deleting any strong bridge of a strong connected graph G may disrupt some 2-edge blocks of G. In this section we study approximation algorithm for the problem of finding a minimum size subset E^* of the strong bridges such that for each strong bridge of G whose removal disrupts some 2-edge blocks of G, the subset E^* contains a strong bridge that has the same effect. We formulate this problem as follows.

Problem 4.7.1
Input: A strongly connected graph $G = (V, E)$.
Task: Find a subset $E^* \subseteq E^{sb}$ of minimum cardinality such that E^* satisfies the following property: For each pair of vertices $v, w \in V$ such that $v \overset{2e}{\rightsquigarrow} w$ in G, if there exists a strong bridge e in G such that the vertices v, w do not lie in the same 2-edge block of $G \setminus \{e\}$, then there is a strong bridge $e' \in E^*$ such that the vertices v, w are not in the same 2-edge block of $G \setminus \{e'\}$.

The decision version of this problem can be formulated as follows. Given a strongly connected graph $G = (V, E)$ and an integer r. Does G have a subset $E^* \subseteq E^{sb}$ that satisfies the following:

1. E^* has at most r elements.
2. For every pair of distinct vertices $v, w \in V$ such that $v \overset{2e}{\rightsquigarrow} w$ in G, the subset E^* contains a strong bridge $e' \in E^{sb}$ such that the vertices v, w are not in the same 2-edge block of $G \setminus \{e'\}$ if G has a strong bridge e such that the vertices v, w do not lie in the same 2-edge block of $G \setminus \{e\}$.

Lemma 4.7.2
The decision version of Problem 4.7.1 is NP-complete

Proof. The proof is similar to the proof of Lemma 4.1.2. □

Now we show that there is a $(1+2\ln n)$ approximation algorithm for Problem 4.7.1 by reducing this problem to the well known minimum set cover problem, defined as follows.

Minimum set cover problem
Input: A set U of l elements, a set X of subsets of U such that $\bigcup_{S\in X} S = U$.
Task: Find a minimum cardinality subset $X^* \subseteq X$ such that $\bigcup_{S\in X^*} S = U$.

Karp [Kar72] showed that the minimum set cover problem is NP-complete. Johnson [Joh74] gave a $(1 + \ln|U|)$ approximation algorithm for this problem.

Lemma 4.7.3
There is a $(1 + 2\ln n)$ approximation algorithm for Problem 4.7.1.

Proof. Let $G = (V, E)$ be a strongly connected graph. We construct an input (U, X) for the minimum set cover problem as follows. The set U contains an element r_{vw} for each pair of vertices v, w that are in the same 2-edge block of G but do not lie in the same 2-edge block of $G \setminus \{e\}$ for some strong bridge $e \in E^{sb}$. (Note that $r_{vw} = r_{wv}$). For each strong bridge $e \in E^{sb}$, we create a set S_e that contains an element $r_{vw} \in U$ for each pair of vertices $v, w \in V$ such that the vertices v, w lie in the same 2-edge block of G but they are no longer in the same 2-edge block after deleting the strong bridge e from G. Therefore, $S_e \subseteq U$ and $U = \bigcup_{e\in E^{sb}} S_e$. Note that each subset S_e can be obtained by computing the 2-edge blocks of $G \setminus \{e\}$ and comparing these blocks with the 2-edge blocks of G. The 2-edge blocks can be calculated in linear time using the algorithm of Georgiadis et al. [Geo+15a]. Therefore, the input (U, X) can be constructed from G in polynomial time in the size of G. Figure 4.11 illustrates an example for this construction.

Let r be an integer. We now show that there is a set cover X^* with $|X^*| = r$ if and only if there is a subset $E^* \subseteq E^{sb}$ in G with $|E^*| = r$ that satisfies the following: For each pair of vertices $v, w \in V$ with $v \overset{2e}{\leftrightsquigarrow} w$, the subset E^* contains a strong bridge e such that v, w do not lie in the same 2-edge block if

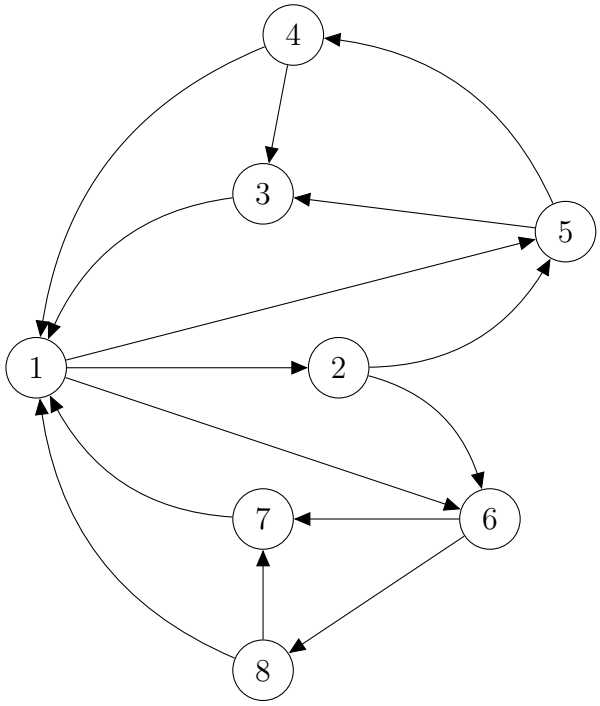


Figure 4.11.: A strongly connected graph $G = (V, E)$ with $E^{sb} = \{(1,2),(5,4),(7,1),(6,8),(3,1)\}$. Since the vertices of $V \setminus \{1,5,6\}$ have indegree or outdegree 1, they do not lie in any 2-edge block of G. The graph G contains only one 2-edge block $C^{2e} = \{1,5,6\}$. For each pair of vertices $v, w \in C^{2e}$ there is a strong bridge $e \in E^{sb}$ such that v, w do not belong to the same 2-edge block of $G \setminus \{e\}$. The input (U, X) of the minimum set cover problem constructed from G:
$S_{12} = \{r_{15}, r_{16}, r_{56}\}$
$S_{54} = \{r_{15}, r_{56}\}$
$S_{71} = \{r_{16}, r_{56}\}$
$S_{68} = \{r_{16}, r_{56}\}$
$S_{31} = \{r_{15}, r_{56}\}$
$U = S_{12} \cup S_{54} \cup S_{71} \cup S_{68} \cup S_{31} = \{r_{15}, r_{16}, r_{56}\}$
Notice that the set $X^* = S_{12}$ is an optimal solution for the minimum set cover problem on this input and corresponds to the set $E^* = \{(1,2)\}$, which is an optimal solution for Problem 4.7.1 on the input G.

there is a strong bridge in G whose removal from G leaves a directed graph in which v, w are not in the same 2-edge block.

"⇒": Let X^* be a set cover with $|X^*| = r$. Let $E^* = \{e \mid S_e \in X^*\}$. By construction, all edges in E^* are strong bridges. Obviously, $|E^*| = r$. Let v, w be a pair of distinct vertices with $v \overset{2e}{\rightsquigarrow} w$ such that v, w do not lie in the same 2-edge block in $G \setminus \{e\}$ for some strong bridge e. Then by construction $r_{vw} \in U$. Since X^* is a set cover, there is some subset $S_{e'} \subseteq X^*$ that contains the element r_{vw}. Consequently, we have $e' \in E^*$.

"⇐": Let E^* be a feasible solution for Problem 4.7.1 such $|E^*| = r$. Suppose that $X^* = \{S_e \mid e \in E^*\}$. It is clear that $|X^*| = |E^*|$. Assume for a contradiction that $\bigcup_{S \in X^*} \neq U$. Then there is a vertex r_{vw} in U such that $r_{vw} \notin \bigcup_{S \in X^*} S$. Since $\{S_e \mid r_{vw} \in S_e\} \cap X^* = \emptyset$, we have $\{e \mid r_{vw} \in S_e\} \cap E^* = \emptyset$. Hence, there is no strong bridge e in the set E^* with the property that the vertices v, w are not in the same 2-edge block of $G \setminus \{e\}$, a contradiction.

Since $|U| < n^2$, Problem 4.7.1 can be approximated within a ratio of $1 + 2 \ln n$. □

Now we define the vertex version of Problem 4.7.1.

Problem 4.7.4
Input: A strongly connected graph $G = (V, E)$ with the 2-directed blocks $C_1^{2d}, C_2^{2d}, \ldots, C_t^{2d}$.
Task: Find a subset $V^* \subseteq V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2d})$ of minimum cardinality such that V^* satisfies the following property:
For each pair of vertices $v, w \in V$ with $v \overset{2}{\rightsquigarrow} w$: there exists a SAP $u \in V^*$ such that v, w are not in the same 2-directed block of $G \setminus \{u\}$ if there is a SAP $v' \in V^{sap} \setminus (\bigcup_{1 \leq i \leq t} C_i^{2d})$ such that v, w do lie in the same 2-directed block of $G \setminus \{v'\}$.

This problem is also NP-hard.

Lemma 4.7.5
The decision version of Problem 4.7.4 is NP-complete.

Proof. The proof is similar to the proof of Lemma 4.3.2. □

There is a $(1 + 2 \ln n)$ approximation algorithm for Problem 4.7.4 since this problem can be reduced to the minimum set cover problem in polynomial time. The proof is similar to the proof of Lemma 4.7.3 (see Figure 4.12).

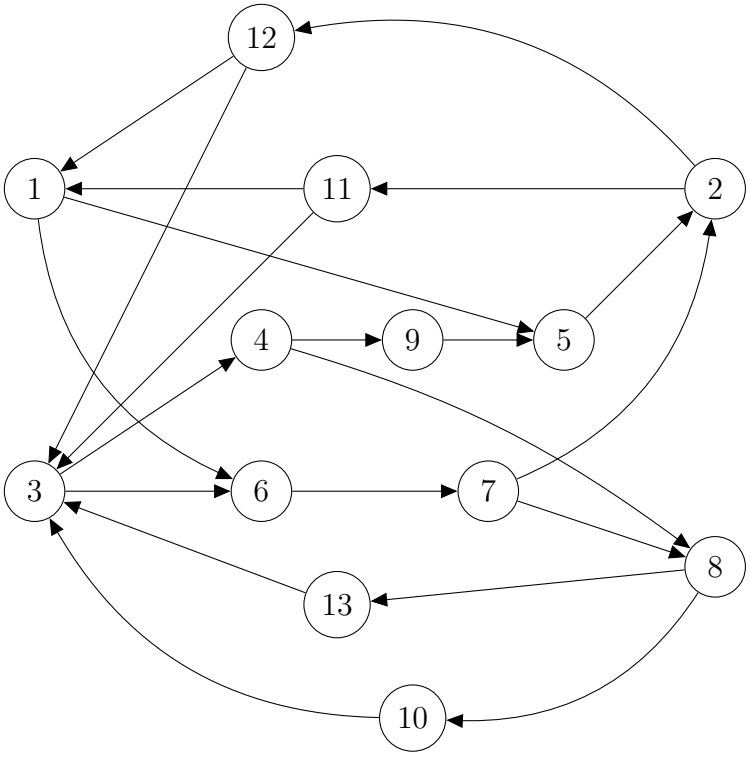


Figure 4.12.: A strongly connected graph $G = (V, E)$. The 2-directed blocks of G are $C_1^{2d} = \{1, 2\}, C_2^{2d} = \{2, 3\}, C_3^{2d} = \{3, 8\}$. Therefore, $V^{sap} \setminus (C_1^{2d} \cup C_2^{2d} \cup C_3^{2d}) = \{4, 5, 6, 7\}$. For each pair of distinct vertices v, w with $v \overset{2}{\rightsquigarrow} w$, there exists a SAP in $V^{sap} \setminus (\bigcup_{1 \leq i \leq 3} C_i^{2d})$ whose removal leaves a directed graph in which v, w are not in the same 2-directed block. The input (U, X) of the minimum set cover problem constructed from G:
$S_4 = \{r_{23}, r_{38}\}$
$S_5 = \{r_{12}, r_{23}\}$
$S_6 = \{r_{12}, r_{23}, r_{38}\}$
$S_7 = \{r_{12}, r_{23}, r_{38}\}$
$U = S_4 \cup S_5 \cup S_6 \cup S_7 = \{r_{12}, r_{23}, r_{38}\}$
The set $X^* = S_6$ is an optimal solution for the minimum set cover problem on this input and corresponds to the set $V^* = \{6\}$, which is an optimal solution for Problem 4.7.4 on the input G.

5. On testing single connectedness in directed graphs and some related problems

The content of this chapter is joint work with Martin Dietzfelbinger, published in [DJ15].

5.1. Introduction

Let $G = (V, E)$ be a directed graph. We assume that the underlying graphs of all directed graphs considered in this chapter are connected; thus $m \geq n - 1$. The graph G is called *singly-connected* if for every pair of vertices $v, w \in V$ there is at most one simple path from v to w in G. The problem of testing whether or not G is singly-connected was introduced by Cormen et al. in [CLR91, Ex. 23.3-10] and [Cor+09, Ex. 22.3-13]. In [BC93; BC92], Buchsbaum and Carlisle presented an algorithm for solving this problem in $O(n^2)$ time. Khuller described in [Khu99] a similar approach for solving the same problem in $O(n^2)$ time. Karlin [Kar95] (also see [Khu00]) also presented a simple $O(n^2)$ algorithm for solving the problem.

Let $G = (V, E)$ be a directed graph. By $G^c = (V^c, E^c)$ we denote the directed graph which is obtained by contracting every strongly connected component of G into a single vertex. The algorithms from [BC93] and [Khu99] are based on reducing the problem on G to the same problem on the acyclic digraph G^c. We use G^r to denote the digraph obtained from G^c by eliminating all vertices of indegree or outdegree 1 by contraction operations. A vertex x of G^r is called a source if its indegree is 0 and a vertex y of G^r is called a sink if its outdegree is 0. By s and t we denote the number of sources and sinks in G^r, respectively. In this chapter we improve the running time of the algorithms from [BC93],[Khu99] from $O(n^2)$ to $O(s \cdot t + m)$. We also give an example for a graph where $s \cdot t$ is much bigger than m. The question posed by Khuller [Khu99] whether the

problem of testing single connectedness in directed graphs is solvable in linear time remains open.

As mentioned in [BC92], it is clear that a singly-connected graph can be obtained from a directed graph which is not singly-connected by removing edges, but the property of singly-connectivity can be ruined by adding edges. We also show that the problem of finding a minimum cardinality edge subset $C \subseteq E$ (respectively, vertex subset $F \subseteq V$) whose removal from G leaves a singly-connected graph is NP-hard.

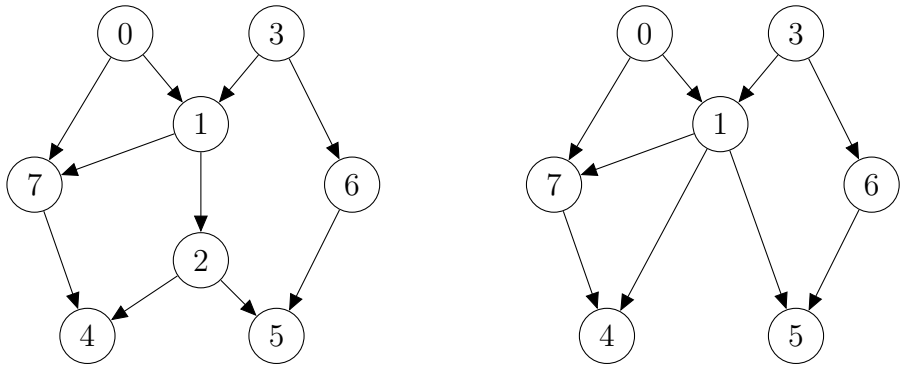
Papers [BC93] and [Khu99] show the existence of a procedure for solving the problem of testing whether or not a directed graph $G = (V, E)$ is singly-connected, which works as follows: It either correctly determines that G is not singly-connected or outputs "Test whether G^c is singly-connected". In the latter case we have that G is singly-connected if and only if G^c is singly-connected. This procedure without testing of G^c has running time $O(m)$. Paper [BC93] shows that the acyclic graph G^c is singly-connected if and only if for every vertex $w \in V^c$ the vertex set of the DFS tree rooted at w has neither cross edges nor forward edges. Testing single connectedness of G^c in this way takes $O(n^2)$ time [BC93; Khu99] since there are n calls to DFS, one for each vertex of G^c, and one can stop as soon as a forward edge or cross edge appears. The procedure leads to the situation that we only have to consider acyclic graphs. We give another reduction, which is to be applied after the procedure described above, can be carried out in time $O(m)$, and yields a reduced graph G^r, with the property that G is singly-connected if and only if G^r is. In G^r no vertex has indegree or outdegree 1.

5.2. Improved handling of acyclic graphs

Assume that the input graph $G = (V, E)$ is acyclic. We propose two types of improvement over the algorithms in [BC93; Khu99] for acyclic graphs:

1. eliminating vertices with indegree or outdegree 1.
2. starting DFS only from sources.

In the first step, called preprocessing step, we modify G as follows. We consider the vertices in bottom-up order. For each vertex $v \in V$ with indegree 1 we shrink u, v, where $(u, v) \in E$, by replacing each edge $(v, w) \in E$ by (u, w) and removing the vertex v and the edge (u, v) from G (see Figure 5.1 for one such contraction). Of course, if v has outdegree 0, it can simply be deleted. Let



(a) G^c (b)

Figure 5.1.: (a) $G = (V, E)$, vertex 2 has indegree 1. (b) The remaining graph after contracting edge $(1, 2)$.

G' be the resulting graph. To G' we apply a similar procedure top down to eliminate all nodes of outdegree 1. The resulting graph is called G^r.

Lemma 5.2.1

(a) G^r can be computed in time $O(m)$.

(b) G is singly-connected if and only if G^r is singly-connected.

Proof (a) We represent the acyclic graph G by adjacency lists in which each vertex $v \in V$ has a circular doubly linked list L_v containing all the successors of v in G. Eliminate all vertices of indegree 1, as follows:

(i) Carry out a depth first search, which will yield a topological order of the vertices of G. Treat the vertices v in reverse topological order (according to increasing postorder numbers) as follows: If v has indegree 1, merge v with its predecessor u by linking L_v into L_u in constant time and deleting v. It is clear this can be done in time $O(m)$. (Actually, the merging itself takes time $O(n)$, only finding vertices with indegree 1 needs $O(m)$ time.)

(ii) Eliminate all vertices of outdegree 1. This is done exactly as in (i), just using the reversed graph of G'.

(b) Each step preserves the property of single connectedness. □
Note that if multiple edges arise in the preprocessing step, then G is not singly-

connected. Check once at the very end, in time $O(m)$, if G^r has multiple edges. If so, return " not singly-connected".

From here on we only consider the reduced graph G^r, in which all non-sources have indegree at least 2 and all non-sinks have outdegree at least 2. In the second step we perform a DFS only for the sources of G^r. The correctness of this step is based on the following lemma.

Lemma 5.2.2
Let $G = (V, E)$ be a directed acyclic graph such that G does not have any multiple edges. Then G is singly-connected if and only if for every source $v \in V$ the vertex set of the DFS tree T_v with root v has only tree edges.

Proof "$\Leftarrow$": Assume that G is not singly-connected. Then by definition, there are two vertices $v, w \in V$ such that there exist at least two simple paths p_1, p_2 from v to w in G. Then there is a vertex u that lies on both paths p_1, p_2 and has two different incoming edges in $p_1 \cup p_2$. There is some source s such that there is a path from s to v in G. Hence the vertex u and all the vertices on $p_1 \cup p_2$ are in T_s. Of course, it is impossible that both edges entering u are tree edges.
"$\Rightarrow$": Any forward or cross edge in a DFS from any vertex immediately proves that G is not singly-connected (see [BC93]). □

Theorem 5.2.3
It can be tested in $O(s \cdot t + m)$ time if G^r is singly-connected.

Proof As we have seen, the transformation from G to G^r takes time $O(m)$. In G^r, each DFS tree T has at most $2t - 1$ vertices since each vertex which is not a leaf in T has outdegree at least 2 and all leaves of T are sinks so there cannot be more than t leaves. Each DFS requires $\Theta(t)$ time because the algorithm stops as soon as a cross edge or a forward edge appears. The total number of DFS-trees is at most s. Therefore, the total running time is $O(s \cdot t + m)$. □
One may ask whether there are singly-connected graphs in which $s \cdot t$ is much bigger than m and in which our algorithm takes much longer than $O(m)$ time. Actually, a well known graph family, the butterfly graphs (or FFT graphs) B_d (see, e.g., [Lei92]), give such an example. They have $n = 2^d(d + 1)$ vertices and $m = 2^{d+1}d$ edges, are singly-connected, and we have $s = t = 2^d$ and hence $s \cdot t = t^2 = 2^{2d} \gg m$.

It is well known that the butterfly graph is singly-connected (see Figure 5.2). Actually, every source is connected to every sink via a unique path. This implies that all DFS calls on sources need time $\Theta(t)$, hence the total time is $\Theta(t^2)$, much bigger than m. (but still smaller than $n^2 = 2^{2d}(d+1)^2$).

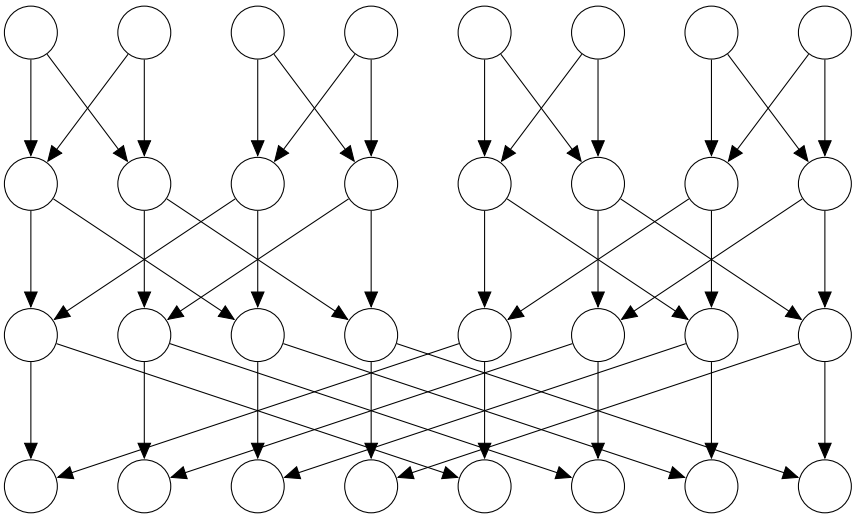


Figure 5.2.: Butterfly graph B_3.

On Butterfly graphs our algorithm takes much more than $O(m)$ time. It remains open to find an algorithm that breaks the $O(s \cdot t)$ bound.

5.3. Optimization problems related to single connectivity

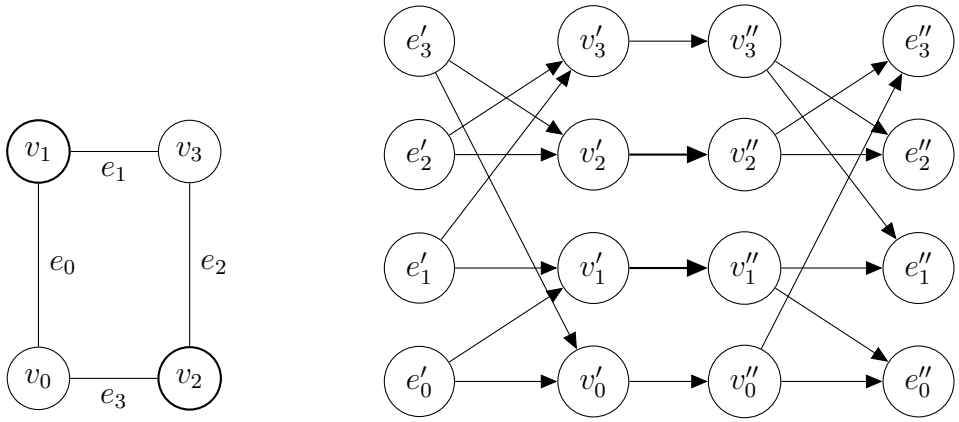
In this section we study two optimization problems related to single connectivity. The first problem is defined as follows. Given a directed graph $G = (V, E)$, find an edge set $C \subseteq E$ of minimum size such that the directed graph $(V, E \setminus C)$ is singly-connected. This problem is denoted by ESC. The second problem is defined as follows. Given a directed graph $G = (V, E)$, find a minimum cardinality vertex set $F \subseteq V$ such that the directed graph $G \setminus F$ obtained from G by removing all the vertices in F and their incident edges is singly-connected. We denote this problem by VSC. We show that VSC and ESC are NP-hard by reducing the vertex cover problem to each of them.

We define the decision version of ESC as follows: Given a directed graph

$G = (V, E)$ and an integer r. Does there exist an edge set $C \subseteq E$ of size at most r such that the directed graph $(V, E \setminus C)$ is singly-connected?

Lemma 5.3.1
The decision version of ESC is NP-complete.

Proof: It is obvious that ESC is in NP. Let $(G = (V, E), l)$ be an instance of the vertex cover problem. We construct an instance $(G' = (V', E'), r)$ of ESC as follows. For each vertex $v \in V$, we add two new vertices v', v'' to V' and a directed edge (v', v'') to E'. Furthermore, for every undirected edge $e = (v, w) \in E$ we add two vertices e', e'' to V' and four directed edges $(e', v'), (e', w'), (v'', e''), (w'', e'')$ to E'. An example is illustrated in Figure 5.3. Clearly, the directed graph G' has $2|V| + 2|E|$ vertices and $|V| + 4|E|$ edges.



(a) $G = (V, E)$, $U = \{v_1, v_2\}$.

(b) $G' = (V', E')$, $C = \{(v'_1, v''_1), (v'_2, v''_2)\}$, the graph $(V', E' \setminus C)$ is singly-connected.

Figure 5.3.: Reducing vertex cover to ESC.

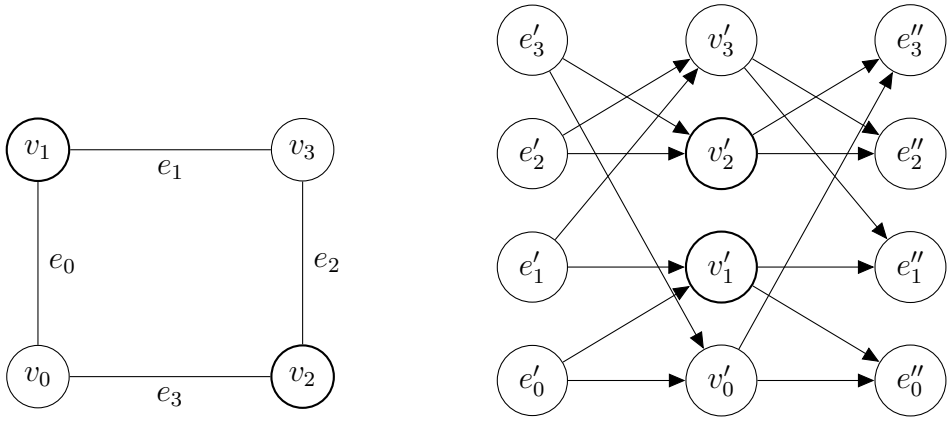
Therefore, G' can be constructed from G in polynomial time. Now we prove that G has a vertex cover of size at most l if and only if G' has an edge set $C \subseteq E'$ of size at most $r = l$ such that $(V', E' \setminus C)$ is singly-connected.
"$\Rightarrow$": Let U be a vertex cover of G such that $|U| \leq l$. Let $C = \{(v', v'') \mid (v', v'') \in E' \text{ and } v \in U\}$. Obviously, $|C| \leq l$. For every pair of vertices $e', e'' \in V'$ which correspond to undirected edge $e = (u, w)$ in G, there exist two simple paths (e', u', u'', e''), (e', w', w'', e'') from e' to e'' in G'. Since $\{u, w\} \cap U \neq \emptyset$, we have $\{(u', u''), (w', w'')\} \cap C \neq \emptyset$. Therefore, there is at most

one simple path from e' to e'' in $(V', E' \setminus C)$. Moreover, for any distinct vertices x, y with $x \neq e'$ or $y \neq e''$, there is at most one simple path from x to y in G'. Each e' is a source, each e'' is a sink and for all $v \in U$ the vertex v' becomes a sink and v'' becomes a source. Consequently, the directed graph $(V', E' \setminus C)$ is singly-connected.
"$\Leftarrow$": Let C be a subset of E' with $|C| \leq l$ such that $(V', E' \setminus C)$ is singly-connected. Let $U = \{v \mid \{(e', v'), (v', v''), (v'', e'')\} \cap C \neq \emptyset \text{ and } v \in V\}$. It is easy to see that $|U| \leq l$. Assume for a contradiction that U is not a vertex cover of G. Then there is an edge (u, w) in G such that $\{u, w\} \cap U = \emptyset$. This implies that $\{(e', u'), (u', u''), (u'', e''), (e', w'), (w', w''), (w'', e'')\} \cap C = \emptyset$. Thus, there are two simple paths (e', u', u'', e'') and (e', w', w'', e'') from e' to e'' in the directed graph $(V', E' \setminus C)$, a contradiction.

Now we define the decision version of VSC. Given a directed graph $G = (V, E)$ and an integer r. Is there a vertex set $F \subseteq V$ of size at most r such that the directed graph $G \setminus F$ is singly-connected. We have the following.

Lemma 5.3.2
The decision version of VSC is NP-complete.

Proof: The proof is similar to the proof of Lemma 5.3.1 (see Figure 5.4).



(a) $G = (V, E)$, $U = \{v_1, v_2\}$ is a vertex cover of G.

(b) $G' = (V', E')$, $F = \{v'_1, v'_2\}$, the graph $G' \setminus F$ is singly-connected.

Figure 5.4.: Reducing vertex cover to VSC.

5.4. Open Problems

The simple $O(nm)$-time algorithm mentioned in the paper of Buchsbaum and Carlisle [BC93] is based on the following fact: A directed graph $G = (V, E)$ is not singly-connected if and only if there is some vertex x such that DFS from x finds a forward edge or a cross edge. Karlin's solution (also see [Khu00]) is based on the observation that we can already conclude that G is not singly-connected if DFS from some x finds a vertex v with two outgoing back edges. Using this speeds up the algorithm to $O(n^2)$. We leave it as an open problem whether Karlin's solution [Kar95] can be improved. Another open problem is whether there are approximation algorithms for the problems described in Section 5.3.

Bibliography

[Als+99] Stephen Alstrup, Dov Harel, Peter W. Lauridsen, and Mikkel Thorup. "Dominators in Linear Time". In: *SIAM J. Comput.* 28.6 (1999), pp. 2117–2132. DOI: 10.1137/S0097539797317263 (cited on pp. 20, 32, 35).

[BC92] Adam L. Buchsbaum and Martin C. Carlisle. *Determining Single Connectivity in Directed Graphs*. Tech. rep. TR-390-92. Princeton University. Aug. 1992. https://www.cs.princeton.edu/research/techreps/TR-390-92 (cited on pp. 95, 96).

[BC93] Adam L. Buchsbaum and Martin C. Carlisle. "Determining Uni-Connectivity in Directed Graphs". In: *Inf. Process. Lett.* 48.1 (1993), pp. 9–12. DOI: 10.1016/0020-0190(93)90261-7 (cited on pp. 95, 96, 98, 102).

[BFL05] Nicolas Beldiceanu, Pierre Flener, and Xavier Lorca. "The tree Constraint". In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, Second International Conference, CPAIOR'05*. Vol. 3524. Lecture Notes in Computer Science. Springer, 2005, pp. 64–78. DOI: 10.1007/11493853_7 (cited on p. 2).

[BR12] R. Balakrishnan and K. Ranganathan. *A Textbook of Graph Theory*. Springer New York, 2012 (cited on pp. 5, 27, 28, 37, 44).

[Buc+08] Adam L. Buchsbaum, Loukas Georgiadis, Haim Kaplan, Anne Rogers, Robert Endre Tarjan, and Jeffery Westbrook. "Linear-Time Algorithms for Dominators and Other Path-Evaluation Problems". In: *SIAM J. Comput.* 38.4 (2008), pp. 1533–1573. DOI: 10.1137/070693217 (cited on pp. 20, 32, 35).

[Car+14] Johannes Carmesin, Reinhard Diestel, M. Hamann, and Fabian Hundertmark. "k-Blocks: A Connectivity Invariant for Graphs". In: *SIAM J. Discrete Math.* 28.4 (2014), pp. 1876–1891. DOI: 10.

1137/130923646. Published previously in arXiv:1305.4557 (cited on pp. 4, 53).

[CLR91] Thomas T. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 1991 (cited on p. 95).

[Cor+09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms (3. ed.)* MIT Press, 2009 (cited on pp. 1, 95).

[CT00] Joseph Cheriyan and Ramakrishna Thurimella. "Approximating Minimum-Size k-Connected Spanning Subgraphs via Matching". In: *SIAM J. Comput.* 30.2 (2000), pp. 528–560. DOI: 10.1137/S009753979833920X (cited on pp. 21, 49, 50).

[Die00] Reinhard Diestel. *Graph Theory, 2nd Edition*. New York: Springer, 2000 (cited on p. 14).

[DJ15] Martin Dietzfelbinger and Raed Jaberi. "On testing single connectedness in directed graphs and some related problems". In: *Information Processing Letters* 115 (2015), pp. 684–688. DOI: 10.1016/j.ipl.2015.04.008 (cited on pp. 7, 95).

[ES80] Y. M. Erusalimskii and G. G. Svetlov. "Bijoin points, bibridges, and biblocks of directed graphs". In: *Cybernetics and Systems Analysis* 16 (1980), pp. 41–44 (cited on pp. 2, 3, 13, 14, 21, 28).

[Eve75] Shimon Even. "An Algorithm for Determining Whether the Connectivity of a Graph is at Least k". In: *SIAM Journal on Computing* 4.3 (1975), pp. 393–396. DOI: 10.1137/0204034. eprint: http://dx.doi.org/10.1137/0204034 (cited on p. 2).

[FHW80] Steven Fortune, John E. Hopcroft, and James Wyllie. "The Directed Subgraph Homeomorphism Problem". In: *Theor. Comput. Sci.* 10 (1980), pp. 111–121. DOI: 10.1016/0304-3975(80)90009-2 (cited on p. 32).

[Fir+12] Donatella Firmani, Giuseppe F. Italiano, Luigi Laura, Alessio Orlandi, and Federico Santaroni. "Computing Strong Articulation Points and Strong Bridges in Large Scale Graphs". In: *Experimental Algorithms - 11th International Symposium, SEA'12*. Vol. 7276. Lecture Notes in Computer Science. Springer, 2012, pp. 195–207. DOI: 10.1007/978-3-642-30850-5_18 (cited on pp. 10, 39).

[FJ81] Greg N. Frederickson and Joseph JáJá. "Approximation Algorithms for Several Graph Augmentation Problems". In: *SIAM J. Comput.* 10.2 (1981), pp. 270–283. DOI: 10.1137/0210019 (cited on p. 45).

[Gab06] Harold N. Gabow. "Using expander graphs to find vertex connectivity". In: *J. ACM* 53.5 (2006), pp. 800–844. DOI: 10.1145/1183907.1183912 (cited on p. 2).

[Gal80] Zvi Galil. "Finding the Vertex Connectivity of Graphs". In: *SIAM Journal on Computing* 9.1 (1980), pp. 197–199. DOI: 10.1137/0209016. eprint: http://dx.doi.org/10.1137/0209016 (cited on p. 2).

[Gav72] Fanica Gavril. "Algorithms for Minimum Coloring, Maximum Clique, Minimum Covering by Cliques, and Maximum Independent Set of a Chordal Graph". In: *SIAM J. Comput.* 1.2 (1972), pp. 180–187. DOI: 10.1137/0201013 (cited on pp. 12, 33).

[Geo+15a] Loukas Georgiadis, Giuseppe F. Italiano, Luigi Laura, and Nikos Parotsidis. "2-Edge Connectivity in Directed Graphs". In: *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'15*. SIAM, 2015, pp. 1988–2005. DOI: 10.1137/1.9781611973730.132. Published previously in Arxiv abs/1407.3041 (cited on pp. 6, 62, 90, 111–116).

[Geo+15b] Loukas Georgiadis, Giuseppe F. Italiano, Luigi Laura, and Nikos Parotsidis. "2-Vertex Connectivity in Directed Graphs". In: *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*. Vol. 9134. Springer, 2015, pp. 605–616. DOI: 10.1007/978-3-662-47672-7_49. previously published in Arxiv abs/1409.6277 (cited on pp. 6, 73, 115).

[Geo10] Loukas Georgiadis. "Testing 2-Vertex Connectivity and Computing Pairs of Vertex-Disjoint *s*-*t* Paths in Digraphs". In: *Automata, Languages and Programming, 37th International Colloquium, ICALP'10, Part I*. Vol. 6198. Lecture Notes in Computer Science. Springer, 2010, pp. 738–749. DOI: 10.1007/978-3-642-14165-2_62 (cited on pp. 2, 44).

[Geo11] Loukas Georgiadis. "Approximating the Smallest 2-Vertex Connected Spanning Subgraph of a Directed Graph". In: *Algorithms - ESA'11 - 19th Annual European Symposium*. Vol. 6942. Lecture Notes in Computer Science. Springer, 2011, pp. 13–24. DOI: 10.1007/978-3-642-23719-5_2 (cited on pp. 20, 21, 44–46, 49, 50).

[GJ79] M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979 (cited on pp. 6, 20).

[GT05] Loukas Georgiadis and Robert Endre Tarjan. "Dominator tree verification and vertex-disjoint paths". In: *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'05*. SIAM, 2005, pp. 433–442 (cited on pp. 44, 47).

[GT12a] Loukas Georgiadis and Robert Endre Tarjan. "Dominator Tree Certification and Independent Spanning Trees". In: *CoRR* abs/1210.8303 (2012) (cited on p. 115).

[GT12b] Loukas Georgiadis and Robert Endre Tarjan. "Dominators, Directed Bipolar Orders, and Independent Spanning Trees". In: *Automata, Languages, and Programming - 39th International Colloquium, ICALP'12*. Vol. 7391. Springer, 2012, pp. 375–386. DOI: 10.1007/978-3-642-31594-7_32 (cited on pp. 10, 48).

[GT14] Andrew V. Goldberg and Robert Endre Tarjan. "Efficient maximum flow algorithms". In: *Commun. ACM* 57.8 (2014), pp. 82–89. DOI: 10.1145/2628036 (cited on pp. 54, 60, 62, 73).

[HKL15] Monika Henzinger, Sebastian Krinninger, and Veronika Loitzenbauer. "Finding 2-Edge and 2-Vertex Strongly Connected Components in Quadratic Time". In: *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*. Vol. 9134. Springer, 2015, pp. 713–724. DOI: 10.1007/978-3-662-47672-7_58. previously published in Arxiv abs/1412.6466 (cited on pp. 2, 3, 111, 116–120).

[HRG00] Monika Rauch Henzinger, Satish Rao, and Harold N. Gabow. "Computing Vertex Connectivity: New Bounds from Old Techniques". In: *J. Algorithms* 34.2 (2000), pp. 222–250. DOI: 10.1006/jagm.1999.1055 (cited on p. 2).

[ILS12] Giuseppe F. Italiano, Luigi Laura, and Federico Santaroni. "Finding strong bridges and strong articulation points in linear time". In: *Theor. Comput. Sci.* 447 (2012), pp. 74–84. DOI: 10.1016/j.tcs.2011.11.011 (cited on pp. 2, 7, 10, 11, 15, 16, 18, 20, 33, 36, 38–40, 44, 45, 47, 48, 51, 53, 62, 73).

[Jab15a] Raed Jaberi. "Computing the 2-blocks of directed graphs". In: *RAIRO - Theor. Inf. and Applic.* 49.2 (2015), pp. 93–119. DOI: 10.1051/ita/2015001 (cited on pp. 6, 111).

[Jab15b] Raed Jaberi. "On computing the 2-vertex-connected components of directed graphs". In: *Discrete Applied Mathematics* (2015). DOI: doi:10.1016/j.dam.2015.10.001 (cited on pp. 13, 28).

[Joh74] David S. Johnson. "Approximation Algorithms for Combinatorial Problems". In: *J. Comput. Syst. Sci.* 9.3 (1974), pp. 256–278. DOI: 10.1016/S0022-0000(74)80044-9 (cited on p. 90).

[Kar72] Richard M. Karp. "Reducibility Among Combinatorial Problems". In: *Proceedings of a symposium on the Complexity of Computer Computations*. The IBM Research Symposia Series. Plenum Press, New York, 1972, pp. 85–103 (cited on pp. 56, 90).

[Kar95] A. Karlin. "Solution to homework 7". CS 421, Winter, Department of Computer Science, University of Washington. 1995 (cited on pp. 95, 102).

[Khu00] Samir Khuller. "Addendum to "An O($|V|^2$) algorithm for single connectedness"". In: *Inf. Process. Lett.* 74.5-6 (2000), p. 263. DOI: 10.1016/S0020-0190(00)00054-5 (cited on pp. 95, 102).

[Khu99] Samir Khuller. "An O(|V|2) algorithm for single connectedness". In: *Inf. Process. Lett.* 72.3-4 (1999), pp. 105–107. DOI: 10.1016/S0020-0190(99)00135-0 (cited on pp. 95, 96).

[KRY94] Samir Khuller, Balaji Raghavachari, and Neal E. Young. "Approximating the Minimum Equivalent Diagraph". In: *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*. ACM/SIAM, 1994, pp. 177–186 (cited on p. 45).

[Lei92] F. Thomson Leighton. *Introduction to Parallel Algorithms and Architectures: Array, Trees, Hypercubes*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1992 (cited on p. 98).

[LM69] Edward S. Lowry and C. W. Medlock. "Object Code Optimization". In: *Commun. ACM* 12.1 (Jan. 1969), pp. 13–22. DOI: 10.1145/362835.362838 (cited on p. 7).

[LMS90] Chung-Lun Li, S. Thomas McCormick, and David Simchi-Levi. "The complexity of finding two disjoint paths with min-max objective function". In: *Discrete Applied Mathematics* 26.1 (1990), pp. 105–115. DOI: 10.1016/0166-218X(90)90024-7 (cited on p. 32).

[LT79] Thomas Lengauer and Robert Endre Tarjan. "A Fast Algorithm for Finding Dominators in a Flowgraph". In: *ACM Trans. Program. Lang. Syst.* 1.1 (1979), pp. 121–141. DOI: 10.1145/357062.357071 (cited on p. 7).

[Lui+15] William Di Luigi, Loukas Georgiadis, Giuseppe F. Italiano, Luigi Laura, and Nikos Parotsidis. "2-Connectivity in Directed Graphs: An Experimental Study". In: *Proceedings of the Seventeenth Workshop on Algorithm Engineering and Experiments, ALENEX'15*. SIAM, 2015, pp. 173–187. DOI: 10.1137/1.9781611973754.15 (cited on pp. 3, 116).

[Mak88] Seishi Makino. "An algorithm for finding all the k-components of a digraph". In: *International Journal of Computer Mathematics* 24.3-4 (1988), pp. 213–221. DOI: 10.1080/00207168808803645. eprint: http://dx.doi.org/10.1080/00207168808803645 (cited on pp. 2, 13, 15).

[Mat77] David W. Matula. "Graph theoretic techniques for cluster analysis algorithms". In: *F. Van Ryzon, ed. Classification and Clustering, Academic Press, New York* (1977), pp. 95–129 (cited on p. 2).

[Mat78] David W. Matula. "k-Blocks and ultrablocks in graphs". In: *J. Comb. Theory, Ser. B* 24.1 (1978), pp. 1–13. DOI: 10.1016/0095-8956(78)90071-0 (cited on p. 15).

[MV88] D. W. Matula and R. V. Vohra. *Calculating the connectivity of a directed graph*. Tech. rep. 386. Institute for Mathematics and Application, University of Minnesota, 1988 (cited on pp. 15, 16).

[Orl13] James B. Orlin. "Max flows in O(nm) time, or better". In: *Symposium on Theory of Computing Conference, STOC'13*. ACM, 2013, pp. 765–774. DOI: 10.1145/2488608.2488705 (cited on pp. 54, 62, 73).

[RS81] J. H. Reif and P. G. Spirakis. *Strong k-connectivity in digraphs and random digraphs*. Tech. rep. TR-25–81. Harvard University, 1981 (cited on p. 15).

[RT75] Donald J. Rose and Robert Endre Tarjan. "Algorithmic Aspects of Vertex Elimination". In: *Proceedings of the 7th Annual ACM Symposium on Theory of Computing, STOC*. ACM, 1975, pp. 245–254. DOI: 10.1145/800116.803775 (cited on pp. 12, 33).

[Tar72] Robert Endre Tarjan. "Depth-First Search and Linear Graph Algorithms". In: *SIAM J. Comput.* 1.2 (1972), pp. 146–160. DOI: 10.1137/0201010 (cited on pp. 2, 4, 14–16, 33, 50).

[Tar76] Robert Endre Tarjan. "Edge-Disjoint Spanning Trees and Depth-First Search". In: *Acta Inf.* 6 (1976), pp. 171–185. DOI: 10.1007/BF00268499 (cited on pp. 51, 52).

[VV00] Santosh Vempala and Adrian Vetta. "Factor 4/3 approximations for minimum 2-connected subgraphs". In: *Approximation Algorithms for Combinatorial Optimization, Third International Workshop, APPROX*. Vol. 1913. Lecture Notes in Computer Science. Springer, 2000, pp. 262–273. DOI: 10.1007/3-540-44436-X_26 (cited on p. 5).

[ZNI03] Liang Zhao, Hiroshi Nagamochi, and Toshihide Ibaraki. "A linear time 5/3-approximation for the minimum strongly-connected spanning subgraph problem". In: *Inf. Process. Lett.* 86.2 (2003), pp. 63–70. DOI: 10.1016/S0020-0190(02)00476-3 (cited on pp. 23, 44–48).

A. Recent developments

In order to place the results reported in this theses in the context of most recent developments and make it easier for the reader to identify this context, we describe the main results of two recent papers.

The algorithm described in Section A.1 for computing 2-edge blocks appeared as TR version in arXiv:1407.3041 in July 2014 and was published in SODA 2015 [Geo+15a]. This work refers to [Jab15a]. In April 2014 a paper which contains the results in Sections 3.1, 3.2, 3.3, and 3.4 was submitted to MFCS but was not accepted. The results in Chapter 3 appeared as TR version in ArXiv:1407.6178 in July 2014 and were published in RAIRO (in 2015) [Jab15a]. The algorithm described in Section A.2 for calculating 2-vccs appeared as TR version in ArXiv:1412.6466 in December 2014 and was published in ICALP 2015 [HKL15].

A.1. Calculating 2-edge blocks in linear time

In [Geo+15a], Georgiadis, Italiano, Laura, and Parotsidis presented a linear time algorithm for calculating the 2-edge blocks of a directed graph. In [Geo+15a], two different $O(nm)$-time algorithms for computing these blocks were given and carefully combined to obtain a linear time algorithm for computing the 2-edge blocks of a directed graph. In this section we briefly describe these algorithms. For details we refere the reader to [Geo+15a].

Let G be a strongly connected graph. For each edge $e \in E \setminus E^{sb}$, the graph $G \setminus \{e\}$ is strongly connected. Let x, y be a pair of distinct vertices such that x, y are not in the same 2-edge block. Then there is some strong bridge $e \in E^{sb}$ such that x, y are in distinct SCCs of $G \setminus \{e\}$. The first $O(nm)$-time algorithm of [Geo+15a], which is described in Algorithm A.1.1, is based on this observation.

Algorithm A.1.1
(from [Geo+15a])
Input: A strongly connected graph $G = (V, E)$.
Output: The 2-edge blocks of G.
1 $B^{curr} \leftarrow \{V\}$.
2 Calculate E^{sb}.
3 **for** each strong bridge $e \in E^{sb}$ **do**
4 Find the SCCs $C_1, C_2, \ldots, C_l$ of $G \setminus \{e\}$.
5 Refine the current 2-edge blocks in B^{curr} by computing $B^{2e}_{cur} \cap C$ for
6 each $B^{2e}_{cur} \in B^{curr}$ and each SCC C of $G \setminus \{e\}$.
7 Output the elements of B^{curr}.

At the beginning, the set of the current 2-edge blocks B^{curr} contains only V in line 1. For each strong bridge e, Algorithm A.1.1 calculates the SCCs $C_1, C_2, \ldots, C_l$ of $G \setminus \{e\}$, gives the vertices of each SCC C_i label i, and refines the current blocks in B^{curr} using bucket sort. The running time of this algorithm is $O(t_{sb}m)$.

Let $G = (V, E)$ be a strongly connected graph and let v be a vertex in G. An edge (x, y) is a *bridge* in the flowgraph $G(v)$ if each path from the start vertex v to the vertex y contains (x, y). The set of dominators of a vertex w in the flowgraph $G^R(v)$ is denoted by $dom^R(w)$.

The authors of [Geo+15a] showed that the 2-edge block C^{2e}_w of G that contains a specific vertex w can be found in linear time using dominator trees and bridges as follows. First the dominator tree $DT(v)$ and the bridges of the flowgraph $G(v)$ are computed. For each bridge (x, y) in $G(v)$, (x, y) is an edge in $DT(v)$. For each bridge (x, y) in $G(v)$, node y in $DT(v)$ is marked. Then the dominator tree $DT^R(v)$ and the bridges of $G^R(v)$ are found. For each bridge (x, y) in $G^R(v)$ node y in $DT^R(v)$ is marked. In [Geo+15a] it was proved that $C^{2e}_w = \{x \mid$ neither $dom(x)$ nor $dom^R(x)$ contains any marked node$\}$.

Next we describe the second $O(nm)$-time algorithm, which is recursive. Let $G = (V, E)$ be a strongly connected graph and let v be a vertex in G. For each bridge (x, y) of $G(v)$ node y in the dominator tree $DT(v)$ is marked. Then all edges $(imd(w), w)$, where w is marked, are removed from $DT(v)$. The removal of these edges decomposes $DT(v)$ into subtrees. The root of each subtree is either v or a marked vertex. This decomposition is called the canonical decomposition of $DT(v)$.

Lemma A.1.2
[Geo+15a] Let C^{2e} be a 2-edge block of a strongly connected graph $G = (V, E)$ and let x, y be two distinct vertices in C^{2e}. Then x, y lie in the same subtree in the canonical decompositions of $DT(v)$ and of $DT^R(v)$.

But two distinct vertices may be not in the same 2-edge block of G and still be in the same subtree in the canonical decompositions of $DT(v)$ and of $DT^R(v)$.

Auxiliary graphs are used in the second $O(nm)$ algorithm and in the final algorithm of [Geo+15a]. Let $T(w)$ be a subtree with root w in the canonical decomposition of $DT(v)$ such that either $w = v$ or w is a marked vertex that is not a leaf in the dominator tree $DT(v)$. The vertices in $T(w)$ that have marked children are called boundary vertices. The auxiliary graph $G_w = (V_w, E_w)$ for the subtree $T(w)$ is defined in the following way. $V_w = V_w^o \cup V_w^a$, where V_w^o is the set of the vertices in the subtree $T(w)$, called ordinary vertices, and V_w^a is a set of auxiliary vertices obtained by contracting the vertices that are not in the subtree $T(w)$ in the following way. All the vertices that are not descendants of w in the dominator tree $DT(v)$ are contracted into $imd(w)$, when $w \neq v$. Let x be a boundary vertex in $T(w)$. For every marked child y of x all the descendants of y in the dominator tree $DT(v)$ are contracted into y. The auxiliary graph G_w is not necessarily a subgraph of G. The authors of [Geo+15a] proved that all auxiliary graphs G_w are strongly connected and can be calculated in $G(m)$ time.

Lemma A.1.3
[Geo+15a] Let $G = (V, E)$ be a strongly connected graph and let x, y be two vertices in G with $x \neq y$. The vertices x, y are in the same 2-edge block of G if and only if they are ordinary vertices in an auxiliary graph G_w and lie in the same 2-edge block of G_w.

This lemma makes it possible to calculate the 2-edge blocks in each auxiliary graph independently from the others. Algorithm A.1.4 shows the second $O(nm)$-time algorithm presented in [Geo+15a].

Algorithm A.1.4
(from [Geo+15a])
Input: A strongly connected graph $G = (V, E)$.
Output: The 2-edge blocks of G.
1 Choose arbitrarily an ordinary vertex $v \in V^o$.

2 Calculate $DT(v)$ and $DT^R(v)$.
3 Calculate b_1, where $b_1 = |\{(x_1, x_2)$ is a bridge in $G(v)$ and x_2 is an
4 ancestor of an ordinary vertex in $DT(v)\}|$.
5 Calculate b_2, where $b_2 = |\{(x_1, x_2)$ is a bridge in $G^R(v)$ and x_2 is an
6 ancestor of an ordinary vertex in $DT^R(v)\}|$.
7 **if** $b_1 = b_2 = 0$ **then**
8 Output the 2-edge block $C_v^{2e} = V^o$.
9 $A \leftarrow G$.
10 **if** $b_2 > b_1$ **then**
11 $A \leftarrow G^R$.
12 Compute the canonical decomposition of the dominator tree of the
13 flowgraph $A(v)$ and the corresponding auxiliary graphs A_w.
14 **for** each A_w containing at least two ordinary vertices **do**
15 Find recursively the 2-edge blocks of A_w.

The authors of [Geo+15a] proved that Algorithm A.1.4 runs in $O(nm)$ time and gave an example for which Algorithm A.1.4 takes $\Omega(nm)$ time. They also noticed that the same strong bridge is used to separate distinct vertices into different blocks in consecutive recursive calls and this strong bridge enters the root of a subtree in the canonical decomposition of a dominator tree. Algorithm A.1.5 shows the linear time algorithm of Georgiadis et al. [Geo+15a] for computing the 2-edge blocks in directed graphs. In this algorithm Algorithm A.1.4 is used but the recursion is stopped at depth 2. Let x, y be a pair of distinct vertices that are not in the same 2-edge block of G but they are not separated when line 12 of Algorithm A.1.5 is reached. Then x, y are ordinary vertices of an auxiliary graph A_q^R calculated at recursion depth 2. In this case it is shown in [Geo+15a] that these vertices are separated by executing lines 3–6 of Algorithm A.1.1 only once since they are in distinct SCCs of $A_q^R \setminus \{(imd_A^R(q), q)\}$.

Algorithm A.1.5
(from [Geo+15a])
Input: A strongly connected graph $G = (V, E)$.
Output: The 2-edge blocks of G.
1 Choose arbitrarily a vertex $v \in V$.
2 Calculate the dominator tree $DT(v)$ of $G(v)$.
3 Compute the canonical decomposition of $DT(v)$ into subtrees $T(w)$
4 and the corresponding auxiliary graphs G_w.
5 **for** every auxiliary graph $A = G_w$ **do**

```
6       Calculate the dominator tree DT_A^R(w) of A^R(w). The parent of
7       q ≠ w in DT_A^R(q) is denoted by imd_A^R(q).
8       Calculate the canonical decomposition of DT_A^R(w) into subtrees
9       T_A^R(q) and the corresponding auxiliary graphs A_q^R with q ≠ w.
10      C_w^{2e} ← the set of the ordinary vertices of T_A^R(w), where C_w^{2e} is
11      the 2-edge block that contains the vertex w .
12      for each auxiliary graph A_q^R such that q ≠ w do
13        Find the SCCs of A_q^R \ {(imd_A^R(q), q)}.
14        for each SCC C of A_q^R \ {(imd_A^R(q), q)} do
16          C_y^{2e} ← the set of the ordinary vertices in C, where y is an
17          ordinary vertex in C and C_y^{2e} is the 2-edge block containing y.
18    Output the 2-edge blocks.
```

Georgiadis, Italiano, Laura, and Parotsidis gave algorithms for computing the 2-strong blocks and the 2-directed blocks of a directed graph in linear time in [Geo+15b]. (A Full version appeared as TR version in ArXiv:1409.6277 in September 2014). Their algorithms follow the high level approach presented in [Geo+15a] for identifying 2-edge blocks. However, their algorithms in [Geo+15b] are more complicated since many difficulties must be handled when following the approach of [Geo+15a], for example, the canonical decomposition, which is used in [Geo+15a] to obtain auxiliary graphs, does not work for computing 2-strong blocks and 2-directed blocks. The graph must be decomposed according to its SAPs. Therefore, the algorithms of [Geo+15b] use a different and more complicated decomposition.

Let $G = (V, E)$ be a directed graph and let x, y be a pair of distinct vertices. Georgiadis et al. [Geo+15b] also showed that the 2-strong blocks can be stored in a data structure of $O(n)$ space. Using this data structure it can be tested in constant time whether x, y are in the same 2-strong block of G. If the vertices x, y do not lie in the same 2-strong block of G, the data structure can report in constant time a SAP whose removal from G leaves x, y in distinct SCCs. In [Geo+15b] it is also shown how to report in $O(1)$ time a SAP or a strong bridge whose removal from G leaves x, y in distinct SCCs when x, y are not in the same 2-directed block.

Using independent spanning trees [GT12a], Georgiadis et al. [Geo+15a; Geo+15b] proved that a directed subgraph G^* of a directed graph G such that the graphs G, G^* have the same 2-blocks and G^* contains only $O(n)$ edges can be computed in linear time.

A.2. Computing the 2-vccs in $O(n^2)$ time

A recent experimental study [Lui+15] showed that our new algorithm (Algorithm 2.3.2) for computing the 2-vccs of a directed graph performs well in practice. Moreover, the authors of [Lui+15] presented a refined version of our algorithm for calculating 2-vccs in directed graphs in $O(nm)$ time. Their experimental study [Lui+15] showed that their own algorithm performs better than our algorithm (Algorithm 2.3.2) in practice (by factor of around 2).

Henzinger et al. [HKL15] gave algorithms for computing the 2-vccs and the 2-edge-connected components of a directed graph in $O(n^2)$ time. In order to obtain these algorithms they used a hierarchical sparsification technique. The authors of [HKL15] also presented algorithms for finding the k-vertex-connected components in $O(n^3)$ time and the k-edge-connected components in $O(n^2 \log n)$ in a directed graph for any constant k. In this section we briefly describe the algorithm of [HKL15] for calculating the 2-vccs of a directed graph in $O(n^2)$ time. The definition of 2-vccs used in [HKL15] follows [Geo+15a]. The description in this section is based on [HKL15].

Let $G = (V, E)$ be a directed graph. Let U be a subset of V that fulfills one of the following two conditions.

1. For any vertex v of that does not belong to U, there is no path from v to any vertex in U.

2. The vertices of the set U are reachable from the set $V \setminus U$ only through one vertex w.

The set U is called a 2-isolated set. In [HKL15] it is shown that for each 2-vcc C^{2vc} of G, either $C^{2vc} \subseteq U \cup \{w\}$ or $C^{2vc} \subseteq V \setminus U$. Each vertex in U has indegree at most $|U|$. Therefore, if the cardinality of U is small, then the vertices of U have a small indegree. To find a 2-isolated set the algorithm searches in subgraphs of G as described later in this section.

A top SCC (tSCC) of G is a SCC C of G such that there is no edge (u, w) with $u \notin C$ and $w \in C$. A bottom SCC (bSCC) of G is a SCC C of G such that there is no edge (u, w) with $u \in C$ and $w \notin C$. Each directed graph that is not strongly connected has a tSCC C_t and a bSCC C_b with $C_t \cap C_b = \emptyset$. Therefore, in each directed graph that is not strongly connected there exist a tSCC C_t and a bSCC C_b with $\min\{|C_t|, |C_b|\} \leq n/2$. A set C is a tSCC of G if and only if C is a bSCC of G^R. A set C is an almost tSCC of G with respect to a vertex w if C is a tSCC of $G \setminus \{w\}$ and there is at least an edge (w, v) with $v \in C$ in G. Each almost tSCC is a 2-isolated set.

Let $G_l = (V_l, E_l)$ be a subgraph of G. The set of vertices v such that the indegree of v in G_l is less than the indegree of v in G is denoted by $B_{G,l}$. The flowgraph $F_{G,l}(s_{G,l})$ is defined in the following way.

1. If $B_{G,l}$ contains at least two vertices, the flowgraph $F_{G,l}(s_{G,l})$ is obtained from G_l by adding a root $s_{G,l}$ to V_l and by adding an edge $(s_{G,l}, w)$, for each vertex $w \in B_{G,l}$, to the set E_l.

2. If $B_{G,l}$ contains only one vertex w, then $s_{G,l} = w$ and $F_{G,l}(s_{G,l}) = G_l(w)$.

In [HKL15] the following is shown.

1. A subset $C \subseteq V$ that does not contain any vertex of $B_{G,l}$ is a tSCC of the subgraph G_l and the flowgraph $F_{G,l}$, respectively if and only if C is a tSCC of G.

2. A subset $C \subseteq V$ that does not contain any vertex of $B_{G,l}$ is an almost tSCC with respect to vertex w of the subgraph G_l and the flowgraph $F_{G,l}$, respectively if and only if C is an almost tSCC of G.

In the algorithm of [HKL15], the subgraphs $G_j = (V, E_j)$, where $E_j = \{$ the first 2^j incoming edges of $v \mid v \in V\}$ (the incoming edges of each vertex are considered in some arbitrary fixed order) are used to find almost tSCCs that have at most a certain cardinality with the help of non-trivial dominators in $F_{G,j}(s_{G,j})$. Let $\alpha = \min\{\max_{w \in V}\{\text{the indegree of } w\}, \max_{w \in V}\{\text{the outdegree of } w\}\}$.

Lemma A.2.1
[HKL15] Let C be a tSCC of G or an almost tSCC of G with respect to some vertex v such that the set C has at most 2^j vertices, then none of the vertices in C belongs to $B_{G,j}$.

The algorithm of Henzinger et al. [HKL15] for computing 2-vccs uses two procedures. The first one is described in Procedure A.2.2.

Procedure A.2.2
(from [HKL15])
Purpose: Looking for a tSCC, an almost tSCC, a bSCC or an almost bSCC.
Input: A directed graph $G = (V, E)$, and an integer j.
Output: A tSCC, an almost tSCC, a bSCC, an almost bSCC or an empty set.
1 **for** each $A \in \{G, G^R\}$ **do**

```
2       E_j ← {the first 2^j incoming edges of w | w ∈ V}.
3       A_j ← (V, E_j).
4       B_{A,j} ← {w | the indegree of w in A is more than 2^j}.
5       look for a tSCC C in the subgraph A_j such that C ∩ B_{A,j} = ∅.
6       if |C| ≥ 1 then
7         return (C, ∅).
8       build the flowgraph graph F_{A,j}(s_{A,j}).
9       if there is a non-trivial dominator in F_{A,j}(s_{A,j}) then.
10        find a non-trivial dominator w in F_{A,j}(s_{A,j}).
11        identify a tSCC C of A_j \ {w} such that C ∩ B_{A,j} = ∅.
12        return (C, {w}).
13      else if B_{A,j} = {s_{A,j}} then
14        if there is a tSCC in A_j \ B_{A,j} whose size at most |V| − 2 then
15        Find a tSCC C of A_j \ B_{A,j}.
16        return (C, {s_{A,j}}).
17    return (∅, ∅).
```

Let $A \in \{G, G^R\}$. Procedure A.2.2 builds the subgraph A_j. Then it looks for a tSCC C in the subgraph A_j such that $C \cap B_{A,j} = \emptyset$. If the procedure finds such tSCC, it returns this tSCC. Otherwise, Procedure A.2.2 builds the flowgraph $F_{A,j}(s_{A,j})$ and looks for a non-trivial dominator in $F_{A,j}(s_{A,j})$. If it finds a non-trivial dominator w, then it returns a tSCC C of $A_j \setminus \{w\}$ such that $C \cap B_{A,j} = \emptyset$ and the vertex w. In the special case that $B_{A,j}$ contains only one vertex $s_{A,j}$, if there is an tSCC in $A_j \setminus \{s_{A,j}\}$ of size at most $|V \setminus \{s_{A,j}\}| - 1$, the procedure returns a tSCC C of $A_j \setminus \{s_{A,j}\}$ and the vertex $s_{A,j}$.

Lemma A.2.3
[HKL15] Procedure A.2.2 returns a 2-isolated set which is not empty when for some integer $1 \leq j < \log \alpha$ there is in some $A \in \{G, G^R\}$ a tSCC S with $S \cap B_{A,j} = \emptyset$ or an almost tSCC S with respect to some vertex w with with $S \cap B_{A,j} = \emptyset$ and $S \subset V \setminus \{w\}$.

Procedure A.2.4 is the second procedure used in the algorithm of [HKL15].

Procedure A.2.4
(from [HKL15])
Purpose: Looking for a tSCC or an almost tSCC of G.
Input: A directed graph $G = (V, E)$.
Output: A tSCC, an almost tSCC, or an empty set.
1 **if** G is not strongly connected **then**
2 find a tSCC C of G.
3 return $(C, \emptyset)$.
4 **if** G contains SAPs **then**
5 find a SAP w of G.
6 find a tSCC C of $G \setminus \{w\}$.
7 return $(C, \{w\})$.
9 return $(\emptyset, \emptyset)$.

Procedure A.2.4 works as follows. The input of this procedure is a directed graph G. Procedure A.2.4 tests whether G is strongly connected. If G is not strongly connected, Procedure A.2.4 finds a tSCC of G and returns it. If G is strongly connected and contains a SAP, Procedure A.2.4 finds a SAP w of G and a tSCC of $G \setminus \{w\}$ and return them. An empty set is returned when G is strongly connected and has no SAPs, i.e, when G is a 2-vcc.

Algorithm A.2.5 shows the algorithm of Henzinger et al. [HKL15], which uses the procedures described above, for calculating the 2-vccs of a directed graph.

Algorithm A.2.5
(from [HKL15])
Input: A directed graph $G = (V, E)$.
Output: The 2-vccs of G.
1 **for** $j \leftarrow 1$ to $\lceil \log \alpha \rceil - 1$ **do**
2 $(C, U) \leftarrow$ Procedure A.2.2(G, j).
3 **if** $|C| \geq 1$ **then**
4 Recursively compute the 2-vccs of $G[C \cup U]$ and $G[V \setminus C]$, respectively.
5 $(C, U) \leftarrow$ Procedure A.2.4(G).
6 **if** $|C| \geq 1$ **then**
7 Recursively compute the 2-vccs of $G[C \cup U]$ and $G[V \setminus C]$, respectively.
8 **else**
10 Output G.

In line 2 of Algorithm A.2.5, if Procedure A.2.2(G, j) returns a tSCC, an

almost tSCC, a bSCC, or an almost bSCC C (i.e., C is not empty), Algorithm A.2.5 recursively computes the 2-vccs of $G[C \cup U]$ and $G[V \setminus C]$, respectively. Otherwise, j is increased by 1, as long as $j < \log \alpha$.

If Procedure A.2.2(G, j) returns $(\emptyset, \emptyset)$ for each $1 \leq j < \log \alpha$, the algorithm calls Procedure A.2.4(G) in order to look for a tSCC or an almost tSCC in G. If Procedure A.2.4(G) returns (C, U) with $C \neq \emptyset$, then Algorithm A.2.5 recursively computes the 2-vccs of $G[C \cup U]$ and $G[V \setminus C]$, respectively. If Procedure A.2.4(G) returns $(\emptyset, \emptyset)$, then Algorithm A.2.5 outputs G since G is a 2-vcc.

Lemma A.2.6
[HKL15] Algorithm A.2.5 calculates all the 2-vccs of a directed graph and runs in $O(n^2)$ time.

Erklärung

Ich versichere, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberaterinnen oder anderen Personen) in Anspruch genommen. Niemand hat von mir unmittelbar oder mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer Prüfungsbehörde vorgelegt.

Ich bin darauf hingewiesen worden, dass die Unrichtigkeit der vorstehenden Erklärung als Täuschungsversuch bewertet wird und gemäß §7 Absatz 10 der Promotionsordnung den Abbruch des Promotionsverfahrens zur Folge hat.

Ilmenau, den 06. Juli 2015

Raed Jaberi