List Homomorphism Problems for Signed Graphs

Jan Bok
Computer Science Institute, Charles University, Prague, Czech Republic
bok@iuuk.mff.cuni.cz

Richard Brewster
Department of Mathematics and Statistics, Thompson Rivers University, Kamloops, Canada
rbrewster@tru.ca

Tomás Feder
Independent Researcher, Palo Alto, CA, USA
tomas@theory.stanford.edu

Pavol Hell
School of Computing Science, Simon Fraser University, Burnaby, Canada
pavol@cs.sfu.ca

Nikola Jedličková
Department of Applied Mathematics, Charles University, Prague, Czech Republic
jedlickova@kam.mff.cuni.cz

Abstract

We consider homomorphisms of signed graphs from a computational perspective. In particular, we study the list homomorphism problem seeking a homomorphism of an input signed graph (G, σ), equipped with lists $L(v) \subseteq V(H), v \in V(G)$, of allowed images, to a fixed target signed graph (H, π). The complexity of the similar homomorphism problem without lists (corresponding to all lists being $L(v) = V(H)$) has been previously classified by Brewster and Siggers, but the list version remains open and appears difficult. Both versions (with lists or without lists) can be formulated as constraint satisfaction problems, and hence enjoy the algebraic dichotomy classification recently verified by Bulatov and Zhuk. By contrast, we seek a combinatorial classification for the list version, akin to the combinatorial classification for the version without lists completed by Brewster and Siggers. We illustrate the possible complications by classifying the complexity of the list homomorphism problem when H is a (reflexive or irreflexive) signed tree. It turns out that the problems are polynomial-time solvable for certain caterpillar-like trees, and are NP-complete otherwise. The tools we develop will be useful for classifications of other classes of signed graphs, and we mention some follow-up research of this kind; those classifications are surprisingly complex.

2012 ACM Subject Classification Mathematics of computing → Graph theory

Keywords and phrases complexity, dichotomy, graph homomorphism, signed graph

Digital Object Identifier 10.4230/LIPIcs.MFCS.2020.20

Funding The second and fourth authors were supported by their NSERC Canada Discovery Grants. The work of the first and fifth author was also partially supported by the grant SVV–2020–260578.
Jan Bok: The first author received funding from the European Union's Horizon 2020 project H2020-MSCA-RISE-2018: Research and Innovation Staff Exchange and from Charles University Grant Agency project 1580119.
Nikola Jedličková: The fifth author was partially supported by the fourth author's NSERC Canada Discovery Grant, and partially supported by Charles University Grant Agency project 1198419.

1 Motivation

We investigate a problem at the confluence of two popular topics – graph homomorphisms and signed graphs. Their interplay was first considered in an unpublished manuscript of Guenin [12], and has since become an established field of study [19].

45th International Symposium on Mathematical Foundations of Computer Science (MFCS 2020).
Editors: Javier Esparza and Daniel Král'; Article No. 20; pp. 20:1–20:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

We now introduce the two topics separately. In the study of computational aspects of graph homomorphisms, the central problem is one of existence – does an input graph G admit a homomorphism to a fixed target graph H? This is known as the *graph homomorphism problem*. It was shown in [15] that this problem is polynomial-time solvable when H has a loop or is bipartite, and is NP-complete otherwise. This is known as the *dichotomy* of graph homomorphisms (see [16]). Now suppose the input graph G is equipped with lists, $L(v) \subseteq V(H), v \in V(G)$, and we ask if there is a homomorphism f of G to H such that each $f(v) \in L(v)$. This is known as the *graph list homomorphism problem*. This problem also has a dichotomy of possible complexities [9] – it is polynomial-time solvable when H is a so-called bi-arc graph and is NP-complete otherwise. Bi-arc graphs have turned out to be an interesting class of graphs; for instance, when H is a reflexive graph (each vertex has a loop), H is a bi-arc graph if and only if it is an interval graph [8].

These kinds of complexity questions found their most general formulation in the context of constraint satisfaction problems. The Feder-Vardi dichotomy conjecture [10] claimed that every constraint satisfaction problem with a fixed template H is polynomial-time solvable or NP-complete. After a quarter century of concerted effort by researchers in theoretical computer science, universal algebra, logic, and graph theory, the conjecture was proved in 2017, independently by Bulatov [7] and Zhuk [25]. This exciting development focused research attention on additional homomorphism type dichotomies, including ones for signed graphs [4, 6, 11].

The study of signed graphs goes back to [13, 14], and has been most notably investigated in [20, 21, 22, 23, 24], from the point of view of colourings, matroids, or embeddings. Following Guenin, homomorphisms of signed graphs have been pioneered in [5] and [18]. The computational aspects of existence of homomorphisms in signed graphs – given a fixed signed graph (H, π), does an input signed graph (G, σ) admit a homomorphism to (H, π) – were studied in [4, 11], and eventually a complete dichotomy classification was obtained in [6]. Although typically homomorphism problems tend to be easier to classify with lists than without lists (lists allow for recursion to subgraphs), the complexity of the list homomorphism problem for signed graphs appears difficult to classify [2, 6]. If the analogy to (unsigned) graphs holds again, then the tractable cases of the problem should identify an interesting class of signed graphs, generalizing bi-arc graphs. In this paper, we begin the exploration of this concept. It turns out that even for signed trees the classification is complex. We illustrate this by classifying the complexity of the list homomorphism problem when H is a (irreflexive, Theorem 9, or reflexive, Theorem 11) signed tree. The problems are polynomial-time solvable for certain caterpillar-like trees, and are NP-complete otherwise. The tools we develop will be useful for classifications of other classes of signed graphs, and we mention some follow-up research of this kind.

2 Terminology and notation

A *signed graph* is a graph G, with possible loops and multiple edges (at most two loops per vertex and at most two edges between a pair of vertices), together with a mapping $\sigma : E(G) \to \{+, -\}$, assigning a sign ($+$ or $-$) to each edge of G, so that different loops at a vertex have different signs, and similarly for multiple edges between the same two vertices. We denote such a signed graph by (G, σ), and call G its *underlying graph* and σ its *signature*. When the signature name is not needed, we denote the signed graph (G, σ) by $\widehat{G}$ to emphasize that it has a signature even though we do not give it a name.

We remark that we view a loop as a special case of an edge (in which the two endpoints coincide). If we need to distinguish an edge other than a loop, we call it a *non-loop edge*.

We will usually view signs of edges as colours, and call positive edges *blue*, and negative edges *red*. It will be convenient to call a red-blue pair of edges with the same endpoint(s) a *bicoloured edge*; however, it is important to keep in mind that formally they are two distinct edges. By contrast, we call edges that are not part of such a pair *unicoloured*; moreover, when we refer to an edge as blue or red we shall always mean the edge is unicoloured blue or red. The sign of a closed walk consisting of unicoloured edges in $\widehat{G}$ is the product of the signs of its edges. Thus a closed walk of unicoloured edges is *negative* if it has an odd number of negative (red) edges, and *positive* if it has an even number of negative (red) edges; in the case of cycles, we also call a positive cycle *balanced*. We call a signed graph *balanced* if all of its cycles are balanced; we also call a signed graph *anti-balanced* if each cycle has an even number of positive edges. A bicoloured edge can be viewed as a trivial cycle with one red and one blue edge, and hence is neither balanced nor anti-balanced; for us a balanced or anti-balanced signed graph does not have bicoloured edges. There is a symmetry to viewing the signs as colours, in particular $\widehat{G}$ is balanced if and only if $\widehat{G}'$, obtained from $\widehat{G}$ by exchanging the colour of each edge, is anti-balanced.

We call a signed graph $\widehat{H}$ *connected* if the underlying graph H is connected. We call $\widehat{H}$ *reflexive* is each vertex of H has a loop, and *irreflexive* if no vertex has a loop. We call $\widehat{H}$ a *tree* if H, with loops removed, is a tree.

We now define the *switching* operation. This operation can be applied to any vertex of a signed graph and it negates the signs of all its incident non-loop edges. (The signs of loops are unchanged by switching.) We say that two signatures σ_1, σ_2 of a graph G are *switching equivalent* if we can obtain (G, σ_2) from (G, σ_1) by a sequence of switchings. In that case we also say that the two signed graphs (G, σ_1) and (G, σ_2) are switching equivalent. In a very formal way, a signed graph is an equivalence class under the switching equivalence, and we sometimes use the notation $\widehat{G}$ to mean the entire class. It was proved by Zaslavsky [21] that two signatures of G are switching equivalent if and only if they define exactly the same set of negative (or positive) cycles. Thus a balanced signed graph is switching equivalent to a signed graph with all edges blue, and an anti-balanced signed graph is switching equivalent to a signed graph with all edges red.

We now consider homomorphisms of signed graphs. Since signed graphs $\widehat{G}, \widehat{H}$ can be viewed as equivalence classes, a homomorphism of signed graphs $\widehat{G}$ to $\widehat{H}$ should be a homomorphism of one representative (G, σ) of $\widehat{G}$ to one representative (H, π) of $\widehat{H}$. It is easy to see that this definition can be simplified by prescribing any fixed representative (H, π) of $\widehat{H}$. In other words, we now consider mapping all possible representatives (G, σ') to one fixed representative (H, π) of $\widehat{H}$. At this point, a homomorphism f of one concrete (G, σ') to (H, π) is just a homomorphism of the underlying graphs G to H preserving the edge colours. Since there are multiple edges, we can either consider f to be a mapping of vertices to vertices and edges to edges, preserving vertex-edge incidences and edge-colours, as in [19], or simply state that blue edges map to blue or bicoloured edges, red edges map to red or bicoloured edges, and bicoloured edges map to bicoloured edges. We follow the second convention.

▶ **Definition 1.** *We say that a mapping $f : V(G) \to V(H)$ is a* homomorphism *of the signed graph (G, σ) to the signed graph (H, π), written as $f : (G, \sigma) \to (H, \pi)$, if there exists a signed graph (G, σ'), switching equivalent to (G, σ), such that whenever uv is a positive edge in (G, σ'), then (H, π) contains a positive edge joining $f(u)$ and $f(v)$, and whenever uv is a negative edge in (G, σ'), then (H, π) contains a negative edge joining $f(u)$ and $f(v)$.*

There is an equivalent alternate definition (see [19]). A homomorphism of the signed graph (G, σ) to the signed graph (H, π) is a homomorphism f of the underlying graphs G to H, such that for any closed walk W in (G, σ) with only unicoloured edges for which the image walk $f(W)$ has also only unicoloured edges, the sign of $f(W)$ in (H, π) is the same as the sign of W in (G, σ). This definition does not require switching the input graph before mapping it. The equivalence of the two definitions follows from the theorem of Zaslavsky [21] cited above. That result is constructive, and the actual switching required to produce the switching equivalent signed graph (G, σ') can be found in polynomial time [19].

▶ **Lemma 2.** *Suppose (G, σ) and (H, π) are signed graphs, and f is a mapping of the vertices of G to the vertices of H. Then f is a homomorphism of the signed graph (G, σ) to the signed graph (H, π) if and only if f is a homomorphism of the underlying graph G to the underlying graph H, which moreover maps bicoloured edges of (G, σ) to bicoloured edges of (H, π), and for any closed walk W in (G, σ) with only unicoloured edges for which the image walk $f(W)$ has also only unicoloured edges, the signs of W and $f(W)$ are the same.*

Note that each negative closed walk contains a negative cycle, and in particular an irreflexive tree (H, π) has no negative closed walks. Thus if (H, π) is an irreflexive tree, then the condition simplifies to having no negative cycle of (G, σ) mapped to unicoloured edges in (H, π). For reflexive trees, the condition requires that no negative cycle of (G, σ) maps to a positive closed walk in (H, π), and no positive cycle of (G, σ) maps to a negative closed walk.

Let $\widehat{H}$ be a fixed signed graph. The *homomorphism problem* S-Hom($\widehat{H}$) takes as input a signed graph $\widehat{G}$ and asks whether there exists a homomorphism of $\widehat{G}$ to $\widehat{H}$. The formal definition of the list homomorphism problems for signed graphs is very similar.

▶ **Definition 3.** *Let $\widehat{H}$ be a fixed signed graph. The* list homomorphism problem *List-S-Hom($\widehat{H}$) takes as input a signed graph $\widehat{G}$ with lists $L(v) \subseteq V(H)$ for every $v \in V(G)$, and asks whether there exists a homomorphism f of $\widehat{G}$ to $\widehat{H}$ such that $f(v) \in L(v)$ for every $v \in V(G)$.*

We note that when $\widehat{H}$ and $\widehat{H}'$ are switching equivalent signed graphs, then any homomorphism of an input signed graph $\widehat{G}$ to $\widehat{H}$ is also a homomorphism to $\widehat{H}'$, and therefore the problems S-Hom($\widehat{H}$) and S-Hom($\widehat{H}'$), as well as the problems List-S-Hom($\widehat{H}$) and List-S-Hom($\widehat{H}'$), are equivalent.

3 More background and connections to constraint satisfaction

We now briefly introduce the constraint satisfaction problems, in the format used in [10]. A *relational system* G consists of a set $V(G)$ of vertices and a family of relations $R_1, R_2, \dots, R_k$ on $V(G)$. Assume G is a relational system with relations $R_1, R_2, \dots, R_k$ and H a relational system with relations $S_1, S_2, \dots, S_k$, where the arity of the corresponding relations R_i and S_i is the same for all $i = 1, 2, \dots, k$. A *homomorphism* of G to H is a mapping $f : V(G) \to V(H)$ that preserves all relations, i.e., satisfies $(v_1, v_2, \dots) \in R_i \implies (f(v_1), f(v_2), \dots) \in S_i$, for all $i = 1, 2, \dots, k$. The *constraint satisfaction problem* with fixed template H asks whether or not an input relational system G, with the same arities of corresponding relations as H, admits a homomorphism to H.

Note that when H has a single relation S, which is binary and symmetric, then we obtain the graph homomorphism problem referred to at the beginning of Section 1. When H has a single relation S, which is an arbitrary binary relation, we obtain the *digraph homomorphism problem* [1] which is in a certain sense [10] as difficult to classify as the general constraint satisfaction problem. When H has two relations $+, -$, then we obtain a problem that is superficially similar to the homomorphism problem for signed graphs, except that switching is not allowed. This problem is called the *edge-coloured graph homomorphism problem* [3], and it turns out to be similar to the digraph homomorphism problem in that it is difficult to classify [4]. On the other hand, the homomorphism problem for signed graphs [4, 6, 11], seems easier to classify, and exhibits a dichotomy similar to the graph dichotomy classification, see Theorem 5.

For homomorphism of signed graphs, Brewster and Graves introduced a useful construction. The *switching graph* (H^+, π^+) of (H, π) is an edge-coloured graph, which has two vertices v_1, v_2 for each vertex v of (H, π), and in which each edge vw of (H, π) gives rise to edges v_1w_1, v_2w_2 of colour $\pi(vw)$ and edges v_1w_2, v_2w_1 of the opposite colour. The same definition applies also for loops, using $v = w$. Then each homomorphism of the signed graph (G, σ) to the signed graph (H, π) corresponds to a homomorphism of (G, σ), this time viewed as an edge-coloured graph, to the edge-coloured graph (H^+, π^+) and conversely. Indeed, mapping a vertex x of G to v_1 corresponds to mapping x to v without first switching at x, and mapping x to v_2 corresponds to first switching at x and then mapping it to v. (Recall that we agreed to only switch in G, in the definition of a homomorphism of signed graphs.) For list homomorphisms of signed graphs, we can use the same transformation, modifying the lists of the input signed graph. If (G, σ) has lists $L(v), v \in V(G)$, then the new lists $L^+(v), v \in V(G)$, are defined as follows: for any $x \in L(v)$ for $x \in V(H), v \in V(G)$, we place both x_1 and x_2 in $L^+(v)$. It is easy to see that the signed graph (G, σ) has a list homomorphism to the signed graph (H, π) with respect to the lists L if and only if the edge-coloured graph (G, σ) has a list homomorphism to the edge-coloured graph (H^+, π^+) with respect to the lists L^+. The new lists L^+ are *symmetric sets in* H^+, meaning that for any $x \in V(G), v \in V(H)$, we have $x_1 \in L^+(v)$ if and only if we have $x_2 \in L^+(v)$. Thus we obtain the list homomorphism problem for the edge-coloured graph (H^+, π^+), restricted to input instances (G, σ) with lists L that are symmetric in H^+. It is well known that list homomorphism problems can be modeled by constraint satisfaction problems if we replace lists by unary relations. (Each subset $X \subseteq V(H)$ gives rise to a unary relation $R_X = X$ in the fixed system H and imposing the corresponding relation S_X on a vertex v of the input system G causes v to map to X, i.e., it is the same as setting the list $L(v) = X$.) We can similarly transform the above list homomorphism problem for the edge-coloured graph (H^+, π^+), to a constraint satisfaction problem with the template H^* obtained by adding unary relations $R_X = X$, for sets $X \subseteq V(H^+)$ that are symmetric in H^+. We conclude that our problems List-S-Hom($\widehat{H}$) fit into the general constraint satisfaction framework, and therefore it follows from [7, 25] that dichotomy holds for problems List-S-Hom($\widehat{H}$). We therefore ask which problems List-S-Hom($\widehat{H}$) are polynomial-time solvable and which are NP-complete.

The solution of the Feder-Vardi dichotomy conjecture involved an algebraic classification of the complexity pioneered by Jeavons [17]. A key role in this is played by the notion of a polymorphism of a relational structure H. If H is a digraph, then a *polymorphism* of H is a homomorphism f of some power H^t to H, i.e., a function f that assigns to each ordered t-tuple $(v_1, v_2, \ldots, v_t)$ of vertices of H a vertex $f(v_1, v_2, \ldots, v_t)$ such that two coordinatewise adjacent tuples obtain adjacent images. For general templates, all relations must be similarly preserved. A polymorphism of order $t = 3$ is a *majority* if $f(v, v, w) = f(v, w, v) = f(w, v, v) = v$ for all v, w. A *Siggers polymorphism* is a polymorphism of order $t = 4$, if $f(a, r, e, a) = f(r, a, r, e)$ for all a, r, e. One formulation of the dichotomy theorem proved by Bulatov [7] and Zhuk [25] states that the constraint satisfaction problem for the template H is polynomial-time solvable if H admits a Siggers polymorphism, and is NP-complete otherwise. Majority polymorphisms are less powerful, but it is known [10] that if H admits a majority then the constraint satisfaction problem for the template H is polynomial-time solvable. Moreover, we have shown in [9] that a graph H is a bi-arc graph if and only if the associated relational system H^* admits a majority polymorphism. Thus the list homomorphism problem for a graph H with possible loops is polynomial-time solvable if H^* admits a majority polymorphism, and is NP-complete otherwise. A similar result may hold for signed graphs. Since List-S-Hom(H, π) is polynomial-time if $(H, \pi)^*$ admits a majority polymorphism, we ask if it is true that List-S-Hom(H, π) is NP-complete if $(H, \pi)^*$ does not admit a majority polymorphism.

There is a convenient way to think of polymorphisms f of the relational system $(H, \pi)^*$. A mapping f is a polymorphism of $(H, \pi)^*$ if and only if it is a polymorphism of the edge-coloured graph (H^+, π^+) and if, for any symmetric set $X \subseteq V(H^+)$, we have $v_1, v_2, \ldots, v_t \in X$ then also $f(v_1, v_2, \ldots, v_t) \in X$. We call such polymorphisms of (H^+, π^+) *semi-conservative*.

We can apply the dichotomy result of [7, 25] to obtain an algebraic classification.

▶ **Theorem 4.** *For any signed graph (H, π), the problem List-S-Hom(H, π) is polynomial-time solvable if (H^+, π^+) admits a semi-conservative Siggers polymorphism, and is NP-complete otherwise.*

As mentioned above, we ask whether in the theorem one can replace the semi-conservative Siggers polymorphism by a semi-conservative majority polymorphism.

In this paper we focus on seeking a graph theoretic classification, at least for some classes of signed graphs.

One of the reasons that, unlike in other contexts, combinatorial dichotomy classification for list homomorphisms of signed graphs appears harder to obtain than for homomorphisms without lists, may be the fact that in the proofs we have to deal with the switching graph of the input signed graph. Indeed, while for graphs the NP-completeness of the list homomorphism problem for an induced subgraph implies the NP-completeness for the whole graph, this is not true for induced subgraphs of the switching graph (because of the required symmetry of the lists).

4 Basic facts

First we record the fact that when $\widehat{H}$ is a *simple* signed graph, i.e., $\widehat{H}$ contains no bicoloured loops or edges, a dichotomy classification of the complexity of the problems List-S-Hom$(\widehat{H})$ is given in [2].

We also mention the dichotomy classification of the problems S-Hom$(\widehat{H})$ from [6]. We recall that a signed graph $\widehat{G}$ is the *s-core* of a signed graph $\widehat{H}$ if there is a homomorphism $f : \widehat{H} \to \widehat{G}$, and every homomorphism $\widehat{G} \to \widehat{G}$ is a bijection on $V(G)$. (We also remind the reader that a loop counts as an edge.)

▶ **Theorem 5.** *[6] The problem S-Hom$(\widehat{H})$ is polynomial-time solvable if the s-core of $\widehat{H}$ has at most two edges, and is NP-complete otherwise.*

Observe that an instance of the problem S-Hom$(\widehat{H})$ can be also viewed as an instance of List-S-Hom$(\widehat{H})$ with all lists $L(v) = V(H)$, therefore if S-Hom$(\widehat{H})$ is NP-complete, then so is List-S-Hom$(\widehat{H})$. Moreover, if $\widehat{H}'$ is an induced subgraph of $\widehat{H}$, then any instance of List-S-Hom$(\widehat{H}')$ can be viewed as an instance of List-S-Hom$(\widehat{H})$ (with the same lists), therefore if the problem List-S-Hom$(\widehat{H}')$ is NP-complete, then so is the problem List-S-Hom$(\widehat{H})$. This yields the NP-completeness of List-S-Hom$(\widehat{H})$ for all signed graphs

$\widehat{H}$ that contain an induced subgraph $\widehat{H}'$ whose s-core has more than two edges. Furthermore, when the signed graph (H, π) is balanced, then we may assume π is positive on all edges, and therefore if the list homomorphism problem for H is NP-complete, then so is the problem List-S-Hom(H, π). In particular, this means that List-S-Hom(H, π) is NP-complete if (H, π) is a balanced signed graph and H is not a bi-arc graph [9].

We have observed that List-S-Hom$(\widehat{H})$ is NP-complete if the s-core of $\widehat{H}$ has more than two edges. Thus we will focus on signed graphs $\widehat{H}$ whose s-cores have at most two edges. There are, however, many complex signed graphs with this property, including, for example, all irreflexive bipartite signed graphs that contain a bicoloured edge, and all signed graphs that contain a bicoloured loop. In this paper we focus on List-S-Hom$(\widehat{H})$ when the underlying graph of $\widehat{H}$ is a reflexive or irreflexive tree. We also study an additional class of irreflexive bipartite signed graphs $\widehat{H}$ in which the unicoloured edges span a Hamiltonian path. We classify the complexity of these graphs; the classification turns out to be surprisingly complex.

We now introduce our basic tool for proving NP-completeness.

▶ **Definition 6.** *Let (U, D) be two walks in $\widehat{H}$ of equal length, say k, with vertices $u = u_0, u_1, \ldots, u_k = v$ and D, with vertices $u = d_0, d_1, \ldots, d_k = v$. We say that (U, D) is a* chain, *provided $uu_1, d_{k-1}v$ are unicoloured edges and $ud_1, u_{k-1}v$ are bicoloured edges, and for each i, $1 \le i \le k-2$, we have*

1. *both u_iu_{i+1} and d_id_{i+1} are edges of $\widehat{H}$ while d_iu_{i+1} is not an edge of $\widehat{H}$, or*
2. *both u_iu_{i+1} and d_id_{i+1} are bicoloured edges of $\widehat{H}$ while d_iu_{i+1} is not a bicoloured edge of $\widehat{H}$.*

▶ **Theorem 7.** *If a signed graph $\widehat{H}$ contains a chain, then List-S-Hom$(\widehat{H})$ is NP-complete.*

5 Irreflexive trees

In this section, $\widehat{H}$ will always be an irreflexive tree. As trees do not have any cycles, they are balanced, and hence we may assume that all edges are either blue or bicoloured.

▶ **Lemma 8.** *Let $\widehat{H}$ be an irreflexive tree. If the underlying graph H contains the graph F_1 in Figure 1, or $\widehat{H}$ contains one of the signed graphs in family $\mathcal{F}$ from Figure 3 as an induced subgraph, then List-S-Hom$(\widehat{H})$ is NP-complete.*

Proof. If the underlying graph H contains the graph F_1 in Figure 1, then H is not a bi-arc graph by [9], whence List-S-Hom$(\widehat{H})$ is NP-complete by the remarks following Theorem 5. If $\widehat{H}$ contains one of the signed graphs in family $\mathcal{F}$ as an induced subgraph, then in each case we apply Theorem 7. The figure lists a chain for each of these forbidden subgraphs. ◀

An irreflexive tree H is a 2-*caterpillar* if it contains a path $P = v_1v_2 \ldots v_k$, such that each vertex of H is either on P, or is a *child* of P, i.e., is adjacent to a vertex on P, or is a *grandchild* of P, i.e., is adjacent to a child of P. We also say that H is a 2-caterpillar *with*

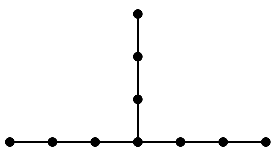

Figure 1 The subgraph F_1.

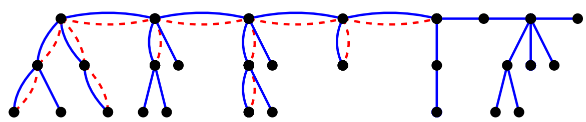

Figure 2 An example of a good 2-caterpillar.

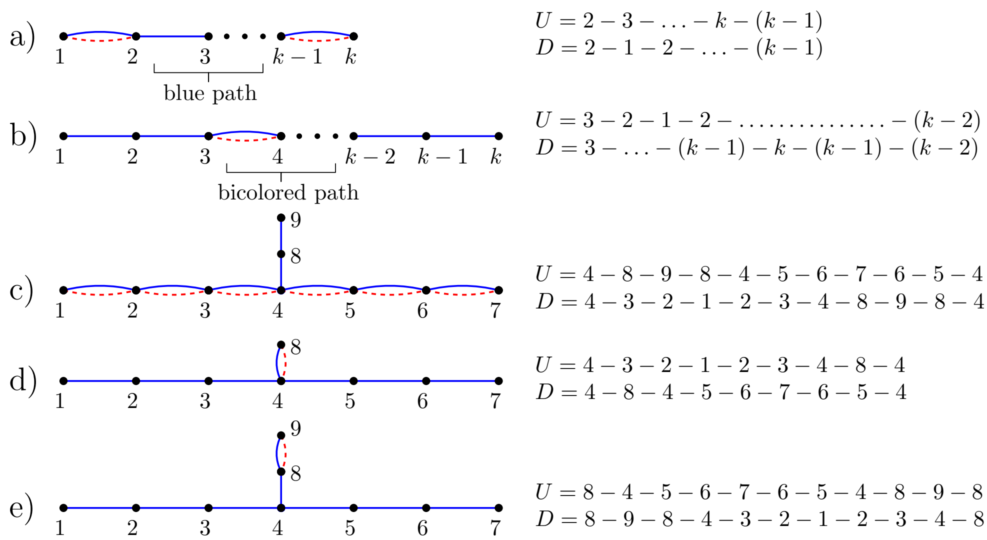

Figure 3 The family $\mathcal{F}$ of signed graphs yielding NP-complete problems, and a chain in each.

respect to the spine P. (Note that the same tree H can be a 2-caterpillar with respect to different spines P.) In such a situation, let $T_1, T_2, \ldots, T_\ell$ be the connected components of $H \setminus P$. Each T_i is a star adjacent to a unique vertex v_j on P. The tree T_i together with the edge joining it to v_j is called a *rooted subtree* of H (with respect to the spine P), and is considered to be rooted at v_j. Note that there can be several rooted subtrees with the same root vertex v_j on the spine, but each rooted subtree at v_j contains a unique child of P (and possibly no grandchildren, or possibly several grandchildren).

If H is a 2-caterpillar with respect to the spine P, and additionally the bicoloured edges of $\widehat{H}$ form a connected subgraph, and there exists an integer d, with $1 \leq d \leq k$, such that:

- all edges on the path $v_1 v_2 \ldots v_d$ are bicoloured, and all edges on the path $v_d v_{d+1} \ldots v_k$ are blue,
- the edges of all subtrees rooted at $v_1, v_2, \ldots, v_{d-1}$ are bicoloured, except possibly edges incident to leaves, and
- the edges of all subtrees rooted at $v_{d+1}, \ldots, v_k$ are all blue,

then we call $\widehat{H}$ a *good* 2*-caterpillar* with respect to $P = v_1 v_2 \ldots v_k$.

The vertex v_d is called the *dividing vertex* of $\widehat{H}$. Note that the subtrees rooted at v_d are not limited by any condition except the connectivity of the subgraph formed by the bicoloured edges. A typical example of a good 2-caterpillar is depicted in Figure 2.

▶ **Theorem 9.** *Let $\widehat{H}$ be an irreflexive tree. If $\widehat{H}$ is a good 2-caterpillar, then* List-S-Hom$(\widehat{H})$ *is polynomial-time solvable. Otherwise, H contains a copy of F_1, or $\widehat{H}$ contains one of the signed graphs in family $\mathcal{F}$ as an induced subgraph, and the problem is NP-complete.*

6 Reflexive trees

We now turn to reflexive trees, and hence in this section, $\widehat{H}$ will always be a reflexive tree. We may have red, blue, or bicoloured loops, but we will assume that all non-loop unicoloured edges are of the same colour (blue or red).

▶ **Lemma 10.** *Let $\widehat{H}$ be a reflexive tree. If the underlying graph H contains the graph F_2 in Figure 4, or $\widehat{H}$ contains one of the signed graphs in family $\mathcal{G}$ depicted in Figure 6 as an induced subgraph, then* List-S-Hom($\widehat{H}$) *is NP-complete.*

Deciding if there exists a list homomorphism (of unsigned graphs) to the graph F_2 is NP-complete [8]. A direct reduction of List-Hom(F_2) to List-S-Hom($\widehat{H}$) as in the proof of Lemma 8 is complicated by the fact that the loops in $\widehat{H}$ can be red, blue, or bicoloured. However, the proof from [8] can itself be adapted to our setting; we skip the details.

A reflexive tree H is a *caterpillar* if it contains a path $P = v_1 \dots v_k$ such that each vertex of H is on P or is adjacent to P. Note that the path P, which we again call the *spine* of H, is not unique, and we sometimes make it explicit by saying that H is a caterpillar *with spine* P. A vertex x not on P is adjacent to a unique neighbour v_i on P, and we call the edge $v_i x$ (with the loop at x) *the subtree rooted at* v_i. A vertex on the spine can have more than one subtree rooted at it. We say that $\widehat{H}$ is a *good caterpillar with respect to the spine* $v_1 \dots v_k$ if the bicoloured edges of $\widehat{H}$ form a connected subgraph, the unicoloured non-loop edges all have the same colour c, and there exists an integer d, with $1 \le d \le k$, such that

- all edges on the path $v_1 v_2 \dots v_d$ are bicoloured, and all edges on the path $v_d v_{d+1} \dots v_k$ are unicoloured with colour c,
- all loops at the vertices $v_1, \dots, v_{d-1}$ and all non-loop edges of the subtrees rooted at these vertices are bicoloured,
- all loops at the vertices $v_{d+1}, \dots, v_k$ and all edges and loops of the subtrees rooted at these vertices are unicoloured with colour c,
- if v_d has a bicoloured loop, then all children of v_d with bicoloured loops are adjacent to v_d by bicoloured edges,
- if v_d has a unicoloured loop of colour c, then all children of v_d have unicoloured loops of colour c, and are adjacent to v_d by unicoloured edges, and
- if $d < k$, then the loops of all children of v_d adjacent to v_d by unicoloured edges also have colour c.

The vertex v_d will again be called the *dividing vertex*. We also say that $\widehat{H}$ is a good caterpillar *with preferred colour* c. Figure 5 shows an example of good caterpillar with preferred colour blue. We emphasize that in the case $d = k$ (not depicted), it is possible (if v_d has a bicoloured loop) that v_d has some children with red loops and some with blue loops, adjacent to v_d by unicoloured edges.

Let $\mathcal{G}$ be the family of signed graphs depicted in Figure 6, together with the family of complementary signed graphs where all unicoloured edges are red, rather than blue, and vice versa. Note that the complementary signed graphs are not switching equivalent to the original signed graphs because switching does not change the colour of loops.

▶ **Theorem 11.** *Let $\widehat{H}$ be a reflexive tree. If $\widehat{H}$ is a good caterpillar, then the problem* List-S-Hom($\widehat{H}$) *is polynomial-time solvable. Otherwise, H contains F_2 from Figure 4, or $\widehat{H}$ contains one of the signed graphs in family $\mathcal{G}$ as an induced subgraph, and the problem is NP-complete.*

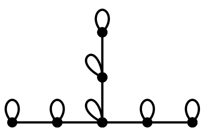

Figure 4 The subgraph F_2.

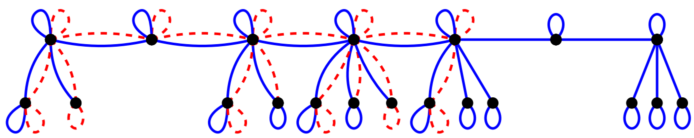

Figure 5 An example of a good caterpillar with preferred colour blue.

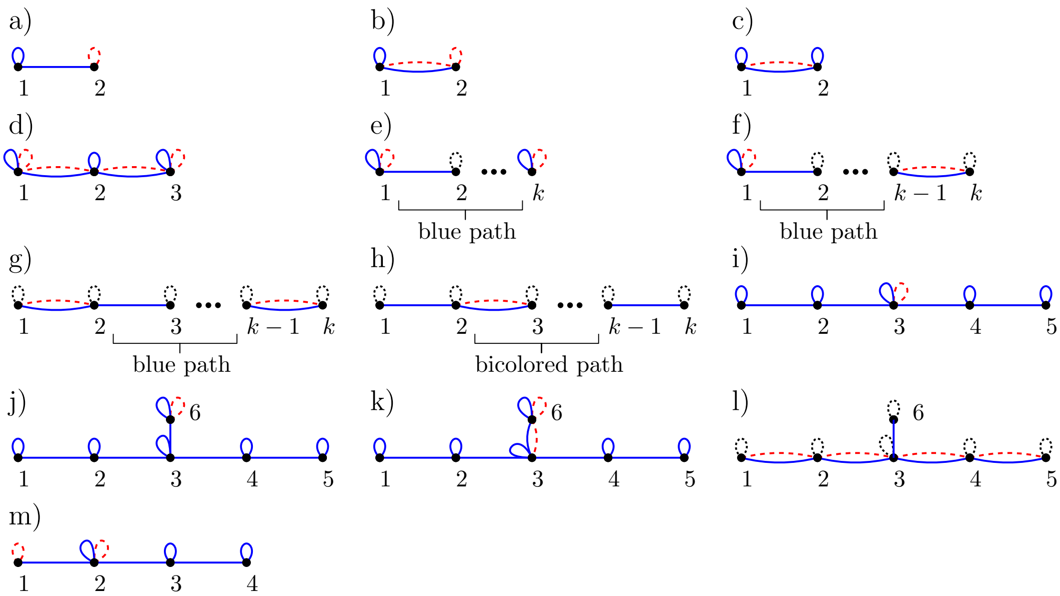

Figure 6 A family $\mathcal{G}$ of signed graphs yielding NP-complete problems. (The solid edges/loops are blue, the dashed edges/loops are red. The dotted loops can be either blue, red or bicoloured.)

We prove the first statement. Thus assume that $\widehat{H}$ is a good caterpillar, with spine $v_1 \dots v_k$ and dividing vertex v_d. By symmetry, we may assume it is a good caterpillar with preferred colour blue. We distinguish three types of rooted subtrees.

- Type T_1: a bicoloured edge $v_i x$ with a bicoloured loop on x;
- Type T_2: a bicoloured edge $v_i x$ with a unicoloured loop on x;
- Type T_3: a blue edge $v_i x$ with a unicoloured loop on x.

A *min ordering* of a graph H is a linear ordering $<$ of the vertices of H, such that for vertices $x < x', y < y'$, if $xy', x'y$ are both edges in H, then xy is also an edge in H. If a graph H admits a min ordering, then the list homomorphism problem for H can be solved in polynomial time by applying the *arc consistency* test, which repeatedly visits edges xy and removes from $L(x)$ any vertex of H not adjacent to some vertex of $L(y)$, and similarly removes from $L(y)$ any vertex of H not adjacent to some vertex of $L(x)$. After the arc consistency test, if there is an empty list, no list homomorphism exists, and if all lists are non-empty, choosing the minimum element of each list, according to $<$, defines a list homomorphism as required [10]. Now suppose $\widehat{H}$ is a good caterpillar with spine $v_1 \dots v_k$ and preferred colour blue. A *special min ordering* of $\widehat{H}$ is a min ordering of the underlying graph H such that for any vertices v_i, x, x' with non-loop edges $v_i x, v_i x'$ we have $x < x'$ whenever (i) the edge $v_i x$ is bicoloured and the edge $v_i x'$ is blue, or (ii) x has a bicoloured loop and x' a unicoloured loop, or (iii) x has a blue loop and x' has a red loop. It is not hard to see that a good caterpillar has a special min ordering.

We now describe our polynomial-time algorithm. We first perform the arc consistency test to check for the existence of a homomorphism of the underlying graphs (G to H), using the special min ordering $<$. Then we also perform the arc consistency test using the bicoloured edges. (Visit bicoloured edges xy of G, remove from $L(x)$ any vertex without a bicoloured edge to some vertex of $L(y)$, and similarly for $L(y)$.) If we get an empty list, there is no list homomorphism. Otherwise, taking again the minima of all lists defines a list homomorphism $f : G \to H$ of the underlying graphs, which now also ensures that f maps bicoloured edges of $\widehat{G}$ to bicoloured edges of $\widehat{H}$. Therefore, by Lemma 2 and the remarks following it, f is also a list homomorphism of signed graphs $\widehat{G} \to \widehat{H}$, unless a negative cycle C of unicoloured edges of $\widehat{G}$ maps to a positive closed walk $f(C)$ of unicoloured edges in $\widehat{H}$, or a positive cycle C of unicoloured edges of $\widehat{G}$ maps to a negative closed walk $f(C)$ of unicoloured edges in $\widehat{H}$. The minimum choices in all lists imply that no vertex x of C can be mapped to an image y with $y < f(x)$. We proceed to modify the images of such cycles C one by one, in the order of increasing smallest vertex in $f(C)$ (in the ordering $<$), until we either obtain a homomorphism of the signed graphs, or we find that no such homomorphism exists.

Let w be the leaf of the last subtree of type T_2 rooted at v_d. We note that if $d < k$, then all edges and loops amongst the vertices that follow w in $<$ are blue, by the properties of a special min ordering. We distinguish three possible cases.

- *At least one vertex y of $f(C)$ satisfies $y \leq w$:*
  The only unicoloured closed walks including y are (red or blue) loops, so f maps the entire cycle C to y. We may remove y from all lists of vertices of C and continue seeking a better homomorphism of the underlying graphs (G to H).
- *All vertices of $f(C)$ follow w in the order $<$ and $d < k$:*
  In this case C is a negative cycle of unicoloured edges. The subgraph of $\widehat{H}$ induced by vertices after w in the order $<$ has only blue edges. Thus there is no homomorphism of signed graphs mapping $\widehat{G} \to \widehat{H}$.
- *All vertices of $f(C)$ follow w in the order $<$ and $d = k$:*
  In this case a fairly complex situation may arise because $f(C)$ can be a closed walk using both red and blue loops, along with blue edges; see below.

We now consider the final case in detail. Since f chooses minimum possible values of images (under $<$), we could only modify f by mapping some vertices of C that were taken by f to a vertex with a blue loop, to vertex with a red loop instead, if lists allow it. We show how to reduce this problem to solving a system of linear equations modulo two, which can then be solved in polynomial time by (say) Gaussian elimination. We begin by considering the pre-image (under f) of all vertices in the subtrees of type T_3 rooted at v_d. We denote by P the set of vertices $v \in V(G)$ with $f(v)$ equal to a vertex with a blue loop and by N the set of vertices $v \in V(G)$ with $f(v)$ equal to a vertex with a red loop. We say that a vertex x of G is a *boundary point* if $f(x) = v_d$. The set of boundary points is denoted by B. Thus the pre-image of the subtrees of type T_3 rooted at v_d is the disjoint union $B \cup P \cup N$. We now focus on the subgraph $\widehat{G}'$ of $\widehat{G}$ induced by $B \cup P \cup N$. A *region* is a connected component of $\widehat{G}' \setminus B$ together with all its boundary points, i.e. between any pair of vertices in a region there is a path with no boundary point as an internal vertex.

Given a region r and boundary points x and y (not necessarily distinct), we construct (possibly several) boolean equations on the corresponding variables, using the same symbols x, y, and r. The variables x, y indicate whether or not the corresponding boundary vertices x and y should be switched before mapping them with f (true corresponds to switching), and the variable r indicates whether the region r will be mapped by f to a blue loop or a red loop (true corresponds to a blue loop). The equations depend of the parity and the sign of walks between the two vertices. If c and d denote parities (even or odd), we say a walk W from x to y in $\widehat{G}'$ is a *(c, d)-walk* if it contains no boundary points other than x and y, the parity of the number of blue edges in W is c, and the parity of the number of red edges in W is d. The equations generated by the (c, d)-walks are as follows. (We indicate the reasoning only in the first case; the other cases are similar.)

- *(odd,odd)-walk:* We add the equation $x = y + 1$. This ensures that exactly one of the boundary vertices has to be switched, in particular x and y must be distinct. The image of the walk must be balanced or anti-balanced (as the whole walk maps to exactly one subtree of type T_3). An even length walk with an odd number of red edges is neither. However, if we switch at exactly one of the endpoints, we can freely map all of the non-boundary points to a blue loop or a red loop.

- *(even,even)-walk:* We add the equation $x = y$.
- *(odd,even)-walk:* We add the equation $x = y + r + 1$.
- *(even,odd)-walk:* We add the equation $x = y + r$.

It is possible that there are several kinds of walks between the same x, y, but we only need to list one of each kind, so the number of equations is polynomial in the size of G. A simple labelling procedure can be used for determining which kinds of walks exist, for given boundary points x and y and a region r. We start at the vertex x, and label its neighbours n_x by the appropriate pairs (c, d), determined by the signs of the edges xn_x. Once a vertex is labelled by a pair (c, d), we correspondingly label its neighbours; a vertex is only given a label (c, d) once even if it is reached with that label several times. Thus a vertex has at most four labels. Any time a vertex receives a new label its neighbours are checked again. The process ends in polynomial time (in the size of the region) as each edge of the region is traversed at most four times. The result is inherent in the labels obtained by y.

Finally, for each region we examine the connected component of the non-boundary vertices. Since the arc consistency procedure was done in the first step of the algorithm, all lists of non-boundary points for a given region are the same. Also, by the ordering $<$, these lists must only contain leaves of v_d. Thus, the non-boundary vertices of the region must map to a single loop. We ensure the choice of the loop is consistent with the lists of each region. If the lists of vertices of some region do not contain a vertex with a red loop, then we add the equation $r = 1$ for the region. Similarly, if the lists do not contain a blue loop, then we add the equation $r = 0$.

Such a system of boolean linear equations can be solved in polynomial time. Also, the system itself is of polynomial size measured by the size of $\widehat{G}$. This completes the proof.

7 Conclusions and future directions

It follows from the dichotomy of constraint satisfaction problems ([7, 25]) that each signed graph $\widehat{H}$ yields a problem List-S-Hom($\widehat{H}$) that is polynomial-time solvable or NP-complete. We have given explicit graph theoretic dichotomy classifications in the case when $\widehat{H}$ is a reflexive or irreflexive tree. The case of general trees (where some vertices have loops and others don't) is a bit more technical, and we will return to it in a journal version of this paper. There we will also illustrate other cases of the classification; even in the irreflexive case the situation is complex, and the tools that we developed here are helpful. In particular, we focus there on graphs in which the unicoloured edges form simple structures, such as spanning cycles, paths, or trees. For an illustration, we state here the results in the case of paths. In this case we may again assume all unicoloured edges are blue.

We say that an irreflexive signed graph $\widehat{H}$ is *path-separable* if the unicoloured edges of $\widehat{H}$ form a Hamiltonian path P in the underlying graph H. In other words, all the edges of the Hamiltonian path P are unicoloured, and all the other edges of $\widehat{H}$ are bicoloured. Recall that the distinction between unicoloured and bicoloured edges is independent of switching, and the Hamiltonian path $P = v_1 v_2 \dots v_n$ is unique, if it exists. A *block* in a path-separable signed graph $\widehat{H}$ is a subpath $v_i v_{i+1} v_{i+2} v_{i+3}$ of P, with the bicoloured edge $v_i v_{i+3}$. A *segment* in $\widehat{H}$ is a maximal subpath $v_i v_{i+1} \dots v_{i+2j+1}$ of P with $j \geq 1$, which has all bicoloured edges $v_{i+e} v_{i+e+o}$ where e is even, $0 \leq e \leq 2j - 2$, and o is odd, $3 \leq o \leq 2j + 1 - e$. (Such a subpath is *maximal* if no such subpath properly contains it.) Note that a segment can be

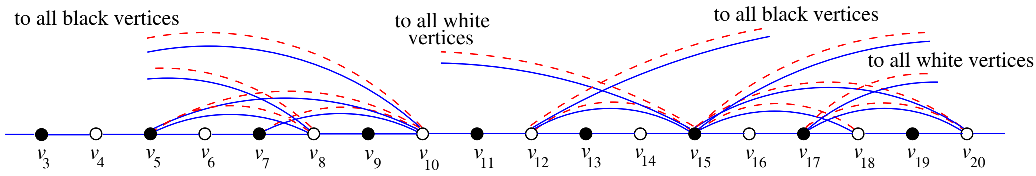

Figure 7 An example of a segmented signed graph of the third kind. The additional bicoloured edges from all white vertices before v_{12} to all black vertices after v_{15} are not shown.

just one block. We say that segments $v_i v_{i+1} \dots v_{i+2j+1}$ and $v_{i'} v_{i'+1} \dots v_{i'+2j'+1}$ with $i < i'$ *avoid* each other if $i' \geq i + 2j + 1$. We say that a segment $v_i v_{i+1} \dots v_{i+2j+1}$ is *right-leaning* if $v_{i+e} v_{i+e+o}$ is a bicoloured edge for all e is even, $0 \leq e \leq 2j-2$, and *all* odd $o \geq 3$; and we say it is *left-leaning* if $v_{i+2j+1-e} v_{i+2j+1-e-o}$ is a bicoloured edge for all e even, $0 \leq e \leq 2j-2$ and all odd $o \geq 3$. We say that a path-separable signed graph $\widehat{H}$ is *segmented* if all segments avoid each other, and one of the following three cases occurs:

1. all segments are left-leaning, or
2. all segments are right-leaning, or
3. a unique segment $v_i v_{i+1} \dots v_{i+2j+1}$ is both left-leaning and right-leaning, all segments preceding it are left-leaning, all segments following it are right-leaning, and there are *additional* bicoloured edges $v_{i-e} v_{i+2j+o}$ with even $e \geq 2$ and odd $o \geq 3$.

See Figure 7: there are three segments, the left-leaning segment $v_5v_6v_7v_8v_9v_{10}$, the left- and right-leaning segment $v_{12}v_{13}v_{14}v_{15}$, and the right-leaning segment $v_{15}v_{16}v_{17}v_{18}v_{19}v_{20}$.

▶ **Theorem 12.** *Let $\widehat{H}$ be a path-separable signed graph. Then* List-S-Hom($\widehat{H}$) *is polynomial-time solvable if $\widehat{H}$ is switching equivalent to a segmented signed graph $\widehat{H}$. Otherwise, the problem is NP-complete.*

References

1 Jørgen Bang-Jensen, Pavol Hell, and Gary MacGillivray. The complexity of colouring by semicomplete digraphs. *SIAM J. Discrete Math.*, 1(3):281–298, 1988.

2 Jan Bok, Richard C. Brewster, Pavol Hell, and Nikola Jedličková. List homomorphisms of signed graphs. In *Bordeaux Graph Workshop*, pages 81–84, 2019.

3 Richard C. Brewster. *Vertex colourings of edge-coloured graphs*. ProQuest LLC, Ann Arbor, MI, 1993. Thesis (Ph.D.)–Simon Fraser University (Canada).

4 Richard C. Brewster, Florent Foucaud, Pavol Hell, and Reza Naserasr. The complexity of signed graph and edge-coloured graph homomorphisms. *Discrete Mathematics*, 340(2):223–235, 2017.

5 Richard C. Brewster and Timothy Graves. Edge-switching homomorphisms of edge-coloured graphs. *Discrete Math.*, 309(18):5540–5546, 2009.

6 Richard C. Brewster and Mark Siggers. A complexity dichotomy for signed h-colouring. *Discrete Mathematics*, 341(10):2768–2773, 2018.

7 Andrei A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *58th Annual IEEE Symposium on Foundations of Computer Science—FOCS 2017*, pages 319–330. IEEE Computer Soc., Los Alamitos, CA, 2017.

8 Tomás Feder and Pavol Hell. List homomorphisms to reflexive graphs. *Journal of Combinatorial Theory, Series B*, 72(2):236–250, 1998.

9 Tomás Feder, Pavol Hell, and Jing Huang. Bi-arc graphs and the complexity of list homomorphisms. *Journal of Graph Theory*, 42(1):61–80, 2003.

10 Tomás Feder and Moshe Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. In *STOC*, pages 612–622, 1993.

11 Florent Foucaud and Reza Naserasr. The complexity of homomorphisms of signed graphs and signed constraint satisfaction. In *Latin American Symposium on Theoretical Informatics*, pages 526–537. Springer, 2014.

12 Bertrand Guenin. Packing odd circuit covers: A conjecture. Manuscript, 2005.

13 Frank Harary. On the notion of balance of a signed graph. *Michigan Math. J.*, 2:143–146 (1955), 1953/54.

14 Frank Harary and Jerald A. Kabell. A simple algorithm to detect balance in signed graphs. *Math. Social Sci.*, 1(1):131–136, 1980/81.

15 Pavol Hell and Jaroslav Nešetřil. On the complexity of H-coloring. *J. Combin. Theory Ser. B*, 48(1):92–110, 1990.

16 Pavol Hell and Jaroslav Nešetřil. *Graphs and homomorphisms*, volume 28 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, Oxford, 2004.

17 Peter Jeavons. On the algebraic structure of combinatorial problems. *Theoret. Comput. Sci.*, 200(1-2):185–204, 1998.

18 Reza Naserasr, Edita Rollová, and Éric Sopena. Homomorphisms of signed graphs. *J. Graph Theory*, 79(3):178–212, 2015.

19 Reza Naserasr, Éric Sopena, and Thomas Zaslavsky. Homomorphisms of signed graphs: An update, 2019. `arXiv:1909.05982`.

20 Thomas Zaslavsky. Characterizations of signed graphs. *J. Graph Theory*, 5(4):401–406, 1981.

21 Thomas Zaslavsky. Signed graph coloring. *Discrete Math.*, 39(2):215–228, 1982.

22 Thomas Zaslavsky. Signed graphs. *Discrete Appl. Math.*, 4(1):47–74, 1982.

23 Thomas Zaslavsky. Is there a matroid theory of signed graph embedding? *Ars Combin.*, 45:129–141, 1997.

24 Thomas Zaslavsky. A mathematical bibliography of signed and gain graphs and allied areas. *Electron. J. Combin.*, 5:Dynamic Surveys 8, 124, 1998. Manuscript prepared with Marge Pratt.

25 Dmitriy Zhuk. A proof of CSP dichotomy conjecture. In *58th Annual IEEE Symposium on Foundations of Computer Science—FOCS 2017*, pages 331–342. IEEE Computer Soc., Los Alamitos, CA, 2017.