RETRACTIONS TO PSEUDOFORESTS

TOMÁS FEDER[†], PAVOL HELL[‡], PETER JONSSON[§], ANDREI KROKHIN[¶], AND GUSTAV NORDH[‖]

Abstract. For a fixed graph H, let $\text{Ret}(H)$ denote the problem of deciding whether a given input graph is retractable to H. We classify the complexity of $\text{Ret}(H)$ when H is a graph (with loops allowed) where each connected component has at most one cycle, i.e., a pseudoforest. In particular, this result extends the known complexity classifications of $\text{Ret}(H)$ for reflexive and irreflexive cycles to general cycles. Our approach is mainly based on algebraic techniques from universal algebra that have previously been used for analyzing the complexity of constraint satisfaction problems.

Key words. Retraction, Computational Complexity, Universal Algebra, Constraint Satisfaction

AMS subject classifications. 05C15, 08A70, 68R10

1. Introduction.

We consider finite, undirected graphs without multiple edges, but with loops allowed. For a graph G, $V(G)$ ($E(G)$) denotes the set of vertices (edges) of G. A graph without loops is called *irreflexive*, a graph in which every vertex has a loop is called *reflexive*, and graphs that are neither irreflexive nor reflexive are called *partially reflexive*.

A *homomorphism* f of a graph G to a graph H is a mapping $f : V(G) \to V(H)$ satisfying the following condition: if $uv \in E(G)$, then $f(u)f(v) \in E(H)$. For a fixed graph H, the *homomorphism problem* $\text{Hom}(H)$ asks whether a graph G admits a homomorphism to H. For instance, if H is K_n (the complete irreflexive graph on n vertices), then $\text{Hom}(H)$ is precisely the n-colouring problem. The complexity of $\text{Hom}(H)$ is known for all graphs [9]; $\text{Hom}(H)$ **NP**-complete if H is irreflexive and non-bipartite, otherwise it is in **P**.

We study a certain generalization of homomorphisms in this article: let G, H be graphs such that H is an induced subgraph of G. A *retraction* r of G to H is a homomorphism of G to H satisfying $r(h) = h$ for every vertex $h \in V(H)$. For a fixed graph H, the *retraction problem* $\text{Ret}(H)$ asks whether a given graph G (having H as an induced subgraph) admits a retraction to H. Retractions and the retraction problem have been intensively studied in graph theory, cf. [10].

In particular, the complexity of $\text{Ret}(H)$ when H is a reflexive cycle, an irreflexive cycle, or a graph on at most four vertices is known, cf. [6, 7, 16]. Hence, what remains to be done in order to complete the classification of $\text{Ret}(H)$ when H is a cycle, is to classify the complexity of $\text{Ret}(H)$ when H is a partially reflexive cycle on 5 or more vertices. In Section 4 we prove that $\text{Ret}(H)$ is **NP**-complete for all partially reflexive cycles H on 5 or more vertices. In Section 5 we extend the classification of $\text{Ret}(H)$ to cover all graphs H in which each connected component has at most one cycle. Such graphs are called pseudoforests and can also be characterized as those graphs that have neither the butterfly (two triangles sharing one vertex) nor the diamond

[†]268 Waverley St., Palo Alto, CA 94301, USA, tomas@theory.stanford.edu

[‡]School of Computing Science, Simon Fraser University, Burnaby, B.C., Canada V5A 1S6 pavol@cs.sfu.ca

[§]Department of Computer and Information Science, Linköpings Universitet, SE-581 83 Linköping, Sweden, petej@ida.liu.se

[¶]School of Engineering and Computing Sciences, South Road, Durham DH1 3LE, UK, andrei.krokhin@durham.ac.uk

[‖]Department of Computer and Information Science, Linköpings Universitet, SE-581 83 Linköping, Sweden, gusno@ida.liu.se

(K_4 with one edge removed) as minors. Our main result is the following complexity classification of $\text{Ret}(H)$ for all pseudoforests H.

- $\text{Ret}(H)$ is **NP**-complete when the looped vertices in a connected component of H induce a disconnected graph, or H contains a cycle on at least 5 vertices, or H contains a reflexive 4-cycle, or H contains an irreflexive 3-cycle.
- $\text{Ret}(H)$ is in **P** for all other pseudoforests H.

Our proof techniques are based on the *algebraic approach* for classifying the complexity of the *constraint satisfaction problem* (Csp) [4, 5, 11]. The Csp problem can be seen as a homomorphism problem on general relational structures as will be explained in Section 2.2. The homomorphism problem to fixed finite target structures $\mathcal{H}$ (denoted $\text{Csp}(\mathcal{H})$) has been intensively studied.

Since $\text{Hom}(H)$ and $\text{Ret}(H)$ are special cases of $\text{Csp}(\mathcal{H})$, the algebraic approach can also be applied to these problems. In fact, Bulatov [2] recently gave a different proof of the dichotomy for $\text{Hom}(H)$ using the algebraic approach, and very recently, Barto et al. [1] used this approach to solve some long standing open questions on the complexity of digraph homomorphisms. We remark that the previous results which we use on the complexity of $\text{Ret}(H)$ from [6, 7, 16] are not proved via the algebraic approach.

Feder and Vardi [8] conjectured that there is a dichotomy (between **P** and **NP**-complete) for the complexity of $\text{Csp}(\mathcal{H})$ (in terms of the relational structures $\mathcal{H}$). This conjecture is still open despite intensive research, although some special cases have been settled, cf. [3]. Feder and Vardi [8] also proved that $\text{Csp}(\mathcal{H})$ has a dichotomy if and only if $\text{Ret}(H)$ has a dichotomy (see also [6, 15] for more information on this connection). Hence, giving a complexity classification of $\text{Ret}(H)$ for all graphs H is probably a very challenging problem. We remark that the reductions between $\text{Csp}(\mathcal{H})$ and $\text{Ret}(H)$, due to Feder and Vardi [8], use connected graphs H with an abundance of cycles.

2. Preliminaries.

2.1. Graphs and retractions. Let G be an arbitrary graph $x \in V(G)$, and $X \subseteq V(G)$. We write $G|_X$ and $G - x$ to denote the subgraphs induced by X and $V(G) \setminus \{x\}$, respectively. We let *loop*(G) denote the set of vertices with loops, i.e., *loop*$(G) = \{x \in V(G) \mid xx \in E(G)\}$ and we let $N_G(x)$ denote the neighborhood of x in G, i.e., $N_G(x) = \{y \in V(G) \mid xy \in E(G)\}$. We will drop the subscript whenever there is no risk of ambiguity. We generalize neighborhoods as follows: $N_G(X) = \bigcup_{y \in X} N(y)$, $N_G^1(x) = N_G(x)$, and $N_G^k(x) = N_G(N_G^{k-1}(x))$ when $k > 1$.

Proposition 2.1 ([16]). *If H is a graph and H' an induced subgraph of H, such that H retracts to H'. Then* $\text{Ret}(H')$ *is polynomial-time reducible to* $\text{Ret}(H)$.

Corollary 2.2. *If H is a graph such that $a, b \in V(H)$ are distinct and $N(a) \subseteq N(b)$, then there is a polynomial-time reduction from* $\text{Ret}(H - a)$ *to* $\text{Ret}(H)$.

Proof. Follows directly from the fact that $N(a) \subseteq N(b)$ implies that H retracts to $H - a$ together with Proposition 2.1. □

Lemma 2.3. *If H is a graph such that $a, b \in V(H)$ are distinct and $N(a) = N(b)$, then* $\text{Ret}(H - a)$ *and* $\text{Ret}(H)$ *are polynomial-time equivalent.*

Proof. The reduction from $\text{Ret}(H-a)$ to $\text{Ret}(H)$ follows from Corollary 2.2. For the other direction, let G be a graph containing H as an induced subgraph. Construct from G a graph G' containing $H - a$ as an induced subgraph by identifying a to b. If r is a retraction from G to H then r', defined as:

- $r'(x) = r(x)$ if $r(x) \neq a$; and
- $r'(x) = b$ if $r(x) = a$

is a retraction from G' to $H - a$. Conversely if r' is a retraction from G' to $H - a$, then r defined as:

- $r(x) = r'(x)$ for all $x \in G'$; and
- $r(a) = a$

is a retraction from G to H. Hence, G retracts to H if and only if G' retracts to $H - a$. □

It has been observed before that when studying the complexity of Ret(H), it is sufficient to consider connected graphs H.

Proposition 2.4 ([16]). *Let H be a graph with connected components $H_1, \ldots, H_n$. Then* Ret(H) *is in* **P** *if* Ret(H_i) *is in* **P** *for all components H_i, and* Ret(H) *is* **NP**-*complete if* Ret(H_i) *is* **NP**-*complete for some component H_i.*

2.2. Constraint satisfaction, retraction, and polymorphisms. For a more extensive treatment we refer the reader to [4, 5]. The *constraint satisfaction problem* (Csp) can be equivalently defined in a number of ways. For our purposes, though, it is convenient to define it as a homomorphism problem. A *vocabulary* is a finite set of relational symbols $R_1, \ldots, R_n$ – each of them have a fixed arity $ar(R_i)$. A *relational structure* over the vocabulary $R_1, \ldots, R_n$ is a structure $\mathcal{H} = (H; R_1^{\mathcal{H}}, \ldots, R_n^{\mathcal{H}})$ where H is a non-empty set (called the *universe* of $\mathcal{H}$) and each $R_i^{\mathcal{H}}$ is a relation on H with arity $ar(R_i)$. Let $\mathcal{G} = (G; R_1^{\mathcal{G}}, \ldots, R_n^{\mathcal{G}})$ and $\mathcal{H} = (H; R_1^{\mathcal{H}}, \ldots, R_n^{\mathcal{H}})$ be relational structures over the vocabulary $R_1, \ldots, R_n$. A *homomorphism* from $\mathcal{G}$ to $\mathcal{H}$ is a mapping $f : G \to H$ such that, for every relation $R^{\mathcal{G}}$ of $\mathcal{G}$ and every tuple $(a_1, \ldots, a_m) \in R^{\mathcal{G}}$, we have $(f(a_1), \ldots, f(a_m)) \in R^{\mathcal{H}}$. A relation of the form $C_a = \{(a)\}$, that is, a unary relation containing only one tuple, is called a *constant relation.* If $\mathcal{H} = (H; R_1, \ldots, R_n)$ is a relational structure, then $\mathcal{H}^c$ denotes the structure $(H; R_1, \ldots, R_n, C_h(h \in H))$.

Let $\mathcal{H}$ be a relational structure over a vocabulary $R_1, \ldots, R_n$. In the constraint satisfaction problem with target structure $\mathcal{H}$, denoted Csp($\mathcal{H}$), the question is, given a structure $\mathcal{G}$ over the same vocabulary, whether there exists a homomorphism from $\mathcal{G}$ to $\mathcal{H}$. Obviously, a graph H can be treated as a relational structure $\mathcal{H} = (V(H); E(H))$. Thus, Hom($H$) and Csp($\mathcal{H}$) (with $\mathcal{H} = (V(H); E(H))$) are equivalent problems. We have the following relation between Csp($\mathcal{H}$) and Ret(H).

Proposition 2.5 ([6]). Ret(H) *and* Csp($\mathcal{H}^c$) *(with $\mathcal{H} = (V(H); E(H))$) are polynomial-time equivalent problems for all graphs H.*

Moreover, adding to a relational structure $\mathcal{H}$ relations derived using certain rules does not change the complexity of the associated Csp [11]. To exemplify this, let Γ be an arbitrary finite set of relations on some finite domain D. Now, let us consider relations derivable from Γ by *primitive positive formulas* (pp-formulas).

Definition 2.6. *The set $\langle \Gamma \rangle$ consists of all relations that can be expressed using*

1. *relations from Γ together with the binary equality relation on D,*
2. *conjunction, and*
3. *existential quantification.*

We say that R is pp-definable in $\mathcal{H} = (H; R_1^{\mathcal{H}}, \ldots, R_n^{\mathcal{H}})$ if $R \in \langle \{R_1^{\mathcal{H}}, \ldots, R_n^{\mathcal{H}}\} \rangle$.

Proposition 2.7 ([11]). *If R is pp-definable in $\mathcal{H}$ and* Csp(R) *is* **NP**-*complete, then* Csp($\mathcal{H}$) *is* **NP**-*complete.*

If R is a unary relation pp-definable in $\mathcal{H}$, then R is called a *subalgebra* of $\mathcal{H}$.

Proposition 2.8 ([2, 4]). *Let H be a graph and $\mathcal{H} = (V(H); E(H))$. Then, for every $v \in V(H)$, $B = N_H^k(v)$ is a subalgebra of $\mathcal{H}^c$ and* Ret(H) *is* **NP**-*complete if* Ret($H|_B$) *is* **NP**-*complete.*

We will now consider *polymorphisms* and their relation to the complexity of Csp($\mathcal{H}$). An n-ary operation f preserves an m-ary relation R (or f is a polymorphism of R, or R is *invariant* under f) if, for any $(a_{11},\ldots,a_{m1}),\ldots,(a_{1n},\ldots,a_{mn}) \in R$, the tuple $(f(a_{11},\ldots,a_{1n}),\ldots,f(a_{m1},\ldots,a_{mn}))$ belongs to R. Given a relational structure $\mathcal{H} = (H; R_1^{\mathcal{H}},\ldots,R_q^{\mathcal{H}})$, if f preserves every relation $R_i^{\mathcal{H}}$ $(1 \leq i \leq q)$ then we say that f is a polymorphism of $\mathcal{H}$. The set of all polymorphism of $\mathcal{H}$ is denoted $Pol(\mathcal{H})$. It is well known that if R is a relation that is pp-definable in $\mathcal{H}$, then $Pol(\mathcal{H}) \subseteq Pol(R)$ [12]. In particular, any subalgebra of $\mathcal{H}$ is preserved by all polymorphisms of $\mathcal{H}$. Recall that an operation $f : D^k \to D$ is said to be idempotent if $f(d,\ldots,d) = d$ for all $d \in D$. Hence, any operation in $Pol(\mathcal{H}^c)$ is idempotent.

Let F be a set of operations on D, B a subset of D and X an equivalence relation on D such that every operation in F preserves B and X. Then $F|_B$ denotes $\{f|_B \mid f \in F\}$ where $f|_B$ is the restriction of f onto B, and $F/_X$ denotes $\{f/_X \mid f \in F\}$ where $f/_X$ is the operation on $D/_X$ defined as $f/_X(d_1/_X,\ldots,d_n/_X) = (f(d_1,\ldots,d_n))/_X$ for any $d_1,\ldots,d_n \in D$.

Finally, we need some information about $Pol(\mathcal{H})$ when $\mathcal{H}$ is a set of relations over some two-element set $\{a,b\} \subseteq D$. To simplify the presentation we assume without loss of generality from now on that D (and $V(H)$) is a subset of $\mathbb{N}$. Let min and max denote the standard binary minimum and maximum operations, let maj denote the majority operation satisfying $\text{maj}(x,x,y) = \text{maj}(x,y,x) = \text{maj}(y,x,x) = y$ for all $x,y \in \{a,b\}$, and define minor to be the minority operation $\text{minor}(x,x,y) = \text{minor}(x,y,x) = \text{minor}(y,x,x) = y$ for all $x,y \in \{a,b\}$. We say that an operation $f : D^k \to D$ is a *projection* if $f(x_1,\ldots,x_k) = x_i$ for all $x_1,\ldots,x_k \in D$.

Theorem 2.9 ([4, 13, 14]). *Let $\mathcal{H}$ be a finite relational structure, B a subalgebra of $\mathcal{H}^c$, and X an equivalence relation on B that is pp-definable in $\mathcal{H}^c$ such that $B/_X$ consists of two elements (equivalence classes). Then, either $((Pol(\mathcal{H}^c))|_B)/_X$ contains projections only and* Csp$(\mathcal{H}^c)$ *is* **NP***-complete, or $((Pol(\mathcal{H}^c))|_B)/_X$ contains a* min, max, majority, *or* minority *operation.*

3. Retraction is hard for graphs with disconnected loops. In this section we prove that Ret(H) is **NP**-complete if there is a connected component H' in H such that the looped vertices in H' induce a disconnected graph.

We first recall the following easy result.

Proposition 3.1. *Given relational structures $\mathcal{G}$ and $\mathcal{H}$ where the universe of $\mathcal{G}$ is $\{g_1,\ldots,g_n\}$ and $HOM(\mathcal{G},\mathcal{H})$ denote the set of all homomorphisms from $\mathcal{G}$ to $\mathcal{H}$, then the relation $S_{\mathcal{G},\mathcal{H}}(g_1,\ldots,g_n) = \{(h(g_1),\ldots,h(g_n)) \mid h \in HOM(\mathcal{G},\mathcal{H})\}$ is pp-definable in $\mathcal{H}$ (i.e., $S_{\mathcal{G},\mathcal{H}} \in \langle\mathcal{H}\rangle$).*

When we are interested in the relation $S_{\mathcal{G},\mathcal{H}}$ we often refer to the relational structure $\mathcal{G}$ as a gadget.

Lemma 3.2. *Let H be a connected graph such that $H|_{loop(H)}$ is not a connected graph, then* Ret(H) *is* **NP***-complete.*

Proof. In this proof we often (implicitly) use the polynomial-time equivalence between Ret(H) and Csp$(\mathcal{H}^c)$ (with $\mathcal{H} = (V(H); E(H))$) from Proposition 2.5. As a rule of thumb, we use the graph H when we are discussing graph properties, and we use the corresponding relational structure $\mathcal{H}^c$ when we are interested in polymorphisms.

To prove this lemma, we will find a subalgebra B and an equivalence relation X, as in Theorem 2.9, such that $((Pol(\mathcal{H}^c))|_B)/_X$ contains none of the four operations listed in Theorem 2.9.

Let d be the minimum distance (in H) between any two vertices from different components of $H|_{loop(H)}$. Fix B to be a minimal (inclusion-wise) subalgebra of $\mathcal{H}^c$ among those that contain two vertices in different components of $H|_{loop(H)}$ at distance d, and also fix two looped vertices a and b in B that witness the above property. Note

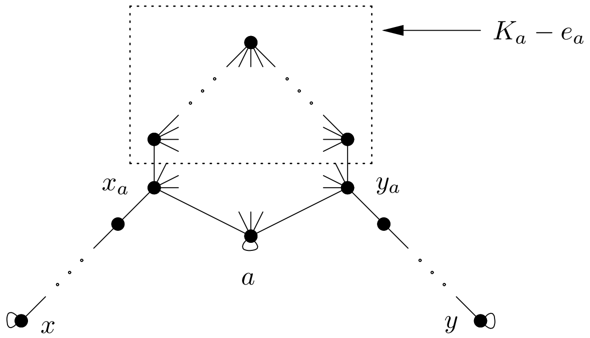

FIG. 3.1. *The gadget G_a.*

that, obviously, B contains only loops because otherwise the loops from B would form a smaller subalgebra with the stated property. Let B_a and B_b denote the sets of elements in B that belong to the connected components of $H|_{loop(H)}$ containing a and b, respectively. Note that $B_a \cap B_b = \emptyset$. We will show that $B = B_a \cup B_b$.

Let K_a be the largest clique in $H|_{V(H)\setminus loop(H)}$ such that there is a vertex $a' \in B_a$ to which every vertex in K_a is adjacent, and every vertex in K_a is at distance $d-1$ from at least one vertex in $B \setminus B_a$. In fact, we can assume that every vertex in K_a is at distance $d-1$ from every vertex in $B \setminus B_a$. Otherwise, there is a vertex $k \in K_a$ that is of distance more that $d-1$ from at least one vertex in B, and hence B can be reduced by taking $B \cap N^{d-1}(k)$. We can without loss of generality assume that a is initially chosen so that $a' = a$.

We will use a gadget construction that can force a vertex in the instance to be mapped only to B_a and B_b. Let e_a be a vertex in K_a and construct a clique on the vertices $V(K_a - e_a) \cup \{a, x_a, y_a\}$, where x_a and y_a are new vertices. Connect x_a and y_a to two new reflexive vertices x and y, respectively, by irreflexive paths of length $d-1$. Force the vertices in $V(K_a - e_a) \cup \{a\}$ to be mapped to the corresponding vertices in H by the constraints $C_i(i)$ for all $i \in V(K_a - e_a) \cup \{a\}$. Finally, force x and y to be mapped to a vertex in B by the constraints $B(x)$ and $B(y)$. Call this gadget G_a (see Figure 3.1 for a pictorial description).

Denote by $\mathcal{H}'^c$ the relational structure $\mathcal{H}^c$ extended with the unary relation B. The property of G_a that we are interested in is that there are homomorphisms h_1, h_2 from G_a to $\mathcal{H}'^c$ such that $h_1(x) = h_2(y) = a$ while $h_1(y), h_2(x)$ are any given elements from B_b, but there is no homomorphism that maps both x and y outside of B_a. The existence of the homomorphisms h_1 and h_2 is easy to verify (using the fact that every vertex in K_a is at distance $d-1$ from every vertex in $B \setminus B_a$) and there can be no homomorphism mapping both x and y to components different from B_a, because the image of clique induced by $V(K_a - e_a) \cup \{x_a, y_a\}$ would have to form an irreflexive clique contradicting the maximality of K_a.

In the same way, we can start from b instead of a and choose a clique K_b and an element b' (which can be assumed to be b), and construct the corresponding gadget G_b, with its own vertices x and y. Now take disjoint union of gadgets G_a and G_b and identify both vertices called x and also both vertices called y. Call this gadget G_{ab}. The same reasoning as above shows that that every homomorphism from G_{ab} to $\mathcal{H}'^c$ must send one of x, y to B_a and the other to B_b, and there exist homomorphisms h_1, h_2 from G_{ab} to $\mathcal{H}'^c$ such that $h_1(x) = h_2(y) = a$ while $h_1(y) = h_2(x) = b$. Consider the relation $S_{G_{ab},\mathcal{H}'^c}(x, y, \ldots)$ and existentially quantify all variables except x. This produces a subalgebra of $\mathcal{H}^c$ which contains a, b and is contained in $B_a \cup B_b$. It now follows from the minimality of B that in fact $B = B_a \cup B_b$.

We now define an equivalence relation on B by constructing a simple gadget G_X consisting of two reflexive vertices r_1 and r_2 that are connected by a reflexive path of length $|H|$ and restricted by the constraints $B(r_1)$ and $B(r_2)$. Recall that any two elements from B_a (or from B_b) are connected by a reflexive path in H, but there is no reflexive path from B_a to B_b. Hence, considering the relation $S_{G_X,\mathcal{H}'^c}(r_1, r_2, \ldots)$ and existentially quantifying over all variables except r_1, r_2 gives us an equivalence relation X on B having equivalence classes B_a and B_b. We will now apply Theorem 2.9 to our $\mathcal{H}^c$, B and X.

Again consider the gadget G_{ab} and the relation $S_{G_{ab},\mathcal{H}'^c}(x, y, \ldots)$ and existentially quantify all variables except x and y, resulting in binary relation R_{ab} which obviously is pp-definable in $\mathcal{H}^c$. Recall from the definition of G_{ab} that R_{ab} contains tuples $(a, b), (b, a)$, but no tuple (c, d) such that $c/_X = d/_X$. Now consider $((Pol(R_{ab}))|_B)/_X$. If $((Pol(R_{ab}))|_B)/_X$ contains a max or min operation $f|_B/_X$, then $f|_B/_X((a/_X, b/_X), (b/_X, a/_X)) = (c/_X, c/_X)$ and R_{ab} would contain a tuple (c, d) such as above. Since R_{ab} is pp-definable in $\mathcal{H}^c$ we have $((Pol(\mathcal{H}^c))|_B)/_X \subseteq ((Pol(R_{ab}))|_B)/_X$, and it follows that $((Pol(\mathcal{H}^c))|_B)/_X$ does not contain any min or max operation.

Consider the clique K_a again, and let $k = |V(K_a)|$. Construct a new clique K on (new) vertices $v_0, v_1, \ldots, v_{k+1}$, where v_0 is looped and the rest are all irreflexive. Join each v_i in $v_1, \ldots v_{k+1}$ to a looped vertex z_i by an irreflexive path of length $d-1$. In order to force the vertices $z_2, \ldots, z_{k+1}$ to be mapped to the same component in $H|_{loop(H)}$ we connect them all by reflexive paths of length $|loop(H)|$. To force all the vertices in $v_0, z_1, z_2, \ldots, z_{k+1}$ to be mapped to a vertex in B we add the constraints $B(v_0), B(z_1), B(z_2), \ldots, B(z_{k+1})$. Call this gadget G_K. There are homomorphisms $h_1, \ldots, h_4$ from G_K to $\mathcal{H}'^c$ such that $(h_1(v_0), h_1(z_1), h_1(z_2)) \in V(B_a) \times V(B_a) \times V(B_a)$, $(h_2(v_0), h_2(z_1), h_2(z_2)) \in V(B_a) \times V(B_b) \times V(B_a)$, $(h_3(v_0), h_3(z_1), h_3(z_2)) \in V(B_a) \times V(B_a) \times V(B_b)$, and $(h_4(v_0), h_4(z_1), h_4(z_2)) \in V(B_b) \times V(B_b) \times V(B_b)$. But there is no homomorphism h from G_K to $\mathcal{H}'^c$ such that $(h(v_0), h(z_1), h(z_2)) \in V(B_a) \times V(B_b) \times V(B_b)$, since the clique $v_1, \ldots, v_{k+1}$ has cardinality $k + 1$ which would contradict the maximality of K_a which has cardinality k.

If we existentially quantify all variables except v_0, z_1, and z_2 in the relation $S_{G_K,\mathcal{H}'^c}(v_0, z_1, z_2, \ldots)$ we get the ternary relation R_K. Now, by the reasoning above R_K contains tuples (a_1, a_2, a_3), (a_4, b_1, a_5), (a_6, a_7, b_2), and (b_3, b_4, b_5) with $a_i/_X = a/_X$ $(1 \leq i \leq 7)$ and $b_i/_X = b/_X$ $(1 \leq i \leq 5)$, but not any tuple (a', b', b'') with $a'/_X = a/_X$ and $b'/_X = b''/_X = b/_X$. Considering $((Pol(R_K))|_B)/_X$, just as in the case of min and max, it is easy to see that $((Pol(R_K))|_B)/_X$ does not contain any majority or minority operation, and thus neither does $((Pol(\mathcal{H}^c))|_B)/_X$. Hence, as a consequence of Theorem 2.9 we get that Ret(H) is **NP**-complete. □

4. Cycles. Here we classify the complexity of Ret(H) when H is a cycle.

Theorem 4.1. *Let H be an n-cycle on vertices $V(H) = \{0, \ldots, n-1\}$, $n \geq 5$, with loops on $\{0, \ldots, m\}$, $m < n-1$. Then,* Ret(H) *is* **NP***-complete.*

We get the following result by combining this theorem with Lemma 3.2 and previously known results for reflexive cycles, irreflexive cycles, and graphs on at most four vertices, cf. Vikas [16].

Corollary 4.2. *Let H be a cycle. Then* Ret(H) *is in* **P** *if H is a 3-cycle having at least one reflexive vertex, or if H is a 4-cycle having at least one irreflexive* *vertex and* $H|_{loop(H)}$ *connected. Otherwise* Ret(H) *is* **NP**-*complete.*

To prove Theorem 4.1, we consider the relational structure $\mathcal{H}^c = (V(H); E(H), C_v\ (v \in V(H)))$ (instead of H). This change of viewpoint is allowed by Proposition 2.5. We prove the result by exhibiting a 2-element subalgebra B of $\mathcal{H}^c$ such that $(Pol(\mathcal{H}^c))|_B$ consists of projections only. By Theorem 2.9 it then follows that Csp$(\mathcal{H}^c)$ is **NP**-complete. The subalgebra we choose is $B = N(n-1) = \{0, n-2\}$; by Proposition 2.8, this is indeed a subalgebra. From Theorem 2.9 we know that $(Pol(\mathcal{H}^c))|_B$ either only consists of projections, or it contains at least one min, max, majority, or minority operation. We proceed by showing that $(Pol(\mathcal{H}^c))|_B$ does not contain any of the four operations above.

Let $\oplus$ and $\ominus$ denote addition and subtraction modulo n, respectively. We need to extend the notion of two vertices being neighbours in a graph to lists of vertices. We say that $(a_1, \ldots, a_n)$ and $(b_1, \ldots, b_n)$ are neighbours in H^n if $a_i \in N_H(b_i)$ for all $1 \le i \le n$.

Lemma 4.3. *If* $f(x, y)$ *is a binary polymorphism of* $\mathcal{H}^c$ *such that* $f(0, n-2) = 0$ *then* $f(x \oplus 2, x) = x \oplus 2$ *for all* x*. Similarly, if* $f(n-2, 0) = 0$ *then* $f(x, x \oplus 2) = x \oplus 2$ *for all* x*.*

Proof. Assume that $f(a \oplus 2, a) = a \oplus 2$ for some a. Since $(a \oplus 1, a \ominus 1)$ is a neighbour of both $(a \oplus 2, a)$ and (a, a) in H^2, it follows that $f(a \oplus 1, a \ominus 1)$ is a neighbour of both $f(a \oplus 2, a)$ and $f(a, a)$ in H. Since $f(a \oplus 2, a) = a \oplus 2$ and $f(a, a) = a$ (because f is idempotent), we have $f(a \oplus 1, a \ominus 1) \in N(a \oplus 2) \cap N(a) = \{a \oplus 1\}$. The lemma follows by induction. □

Lemma 4.4. *If* $f(x, y)$ *is a binary polymorphism of* $\mathcal{H}^c$ *such that* $f(0, n-2) = n-2$ *then* $f(0, x) = x$ *for all* x*. Similarly, if* $f(n-2, 0) = n-2$ *then* $f(x, 0) = x$ *for all* x*.*

Proof. Since $(0, n-1)$ is a neighbour of both $(0, 0)$ and $(0, n-2)$ in H^2, it follows that $f(0, n-1) = n-1$. Assume the lemma is false and let $a \in \{0, \ldots, n-1\}$ be the largest element such that $f(0, a) = a$, but $f(0, a \ominus 1) \neq a \ominus 1$. Since $(0, a)$ and $(0, a \ominus 1)$ are neighbours in H^2, it follows that $f(0, a \ominus 1) \in N(a) \setminus \{a \ominus 1\} \subseteq \{a, a \oplus 1\}$. Consider first the case where $a \ominus 1$ is even and let $k = \frac{a \ominus 1}{2}$. Then $0, a \ominus 1 \in N^k(k)$, but $a, a \oplus 1 \notin N^k(k)$. Since $n \ge 5$ and f preserves $N^k(k)$, it follows that $f(0, a \ominus 1) \notin \{a, a \oplus 1\}$ which is a contradiction. Now, consider the case where $a \ominus 1$ is odd and let $k = \frac{a \ominus 2}{2}$. Then $f(0, a \ominus 1)$ is in $N^{k+1}(k) \subseteq \{n-1, 0, 1, \ldots, a \ominus 1\}$. Note that, by definition, $a \ominus 1$ cannot be any of $0, n-1, n-2$. Hence, we must have $f(0, a \ominus 1) = a \oplus 1$ and $a = n-2$ (and n is even). So, $f(0, n-3) = n-1$, and we have $f(0, n-4) = 0$ because $f(0, n-4)$ must be in $N(f(0, n-3))$ and also in the $(n-4)/2$-neighborhood of $(n-4)/2$. Finally, $f(1, n-3)$ must belong to each of $N(f(0, n-4))$, $N(f(0, n-2))$ and $N^{t-1}(t)$ where $t = (n-2)/2$, but the intersection of these three sets is empty. □

Proof. [Of Theorem 4.1]

<u>Case 1.</u> $f(0, n-2) = f(n-2, 0) = 0$, i.e., f is the min function on $\{0, n-2\}$
By Lemma 4.3, we have $f(m, m \oplus 2) = f(m \oplus 2, m) = m \oplus 2$ (recall that m is the last vertex with a loop). Then $f(m, m \oplus 1) = m \oplus 1$ because it must be a neighbour of both $f(m, m) = m$ and $f(m, m \oplus 2) = m \oplus 2$. Similarly, we have $f(m \oplus 1, m) = m \oplus 1$. Hence, since $(m, m \oplus 1)$ and $(m \oplus 1, m)$ are neighbours in H^2, we have $(m \oplus 1, m \oplus 1) \in E(H)$ which is a contradiction with the fact that $m \oplus 1$ is not looped.

<u>Case 2.</u> $f(0, n-2) = f(n-2, 0) = n-2$, i.e., f is the max function on $\{0, n-2\}$
By Lemma 4.4, we have $f(0, n-1) = f(n-1, 0) = n-1$.

Since $(0, n-1)$ and $(n-1, 0)$ are neighbours, we have $(n-1, n-1) \in E(H)$ which contradicts the fact that $n-1$ is not looped.

<u>Case 3.</u> f is a majority operation on $\{0, n-2\}$.

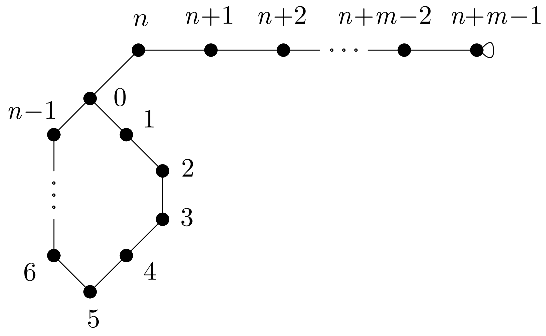

FIG. 5.1. *Balloon:* $B_{n,m}$

Consider the operation $g(x, y) = f(x, x, y)$ on V. This is a polymorphism of $\mathcal{H}^c$. Note that $g(n-2, 0) = f(n-2, n-2, 0) = n-2$. By applying Lemma 4.4 to g, we obtain that $f(x, x, 0) = g(x, 0) = x$ for all x. Consider the operation $g'(x, y) = f(x, y, 0)$. This operation is a polymorphism of $\mathcal{H}^c$ because it is idempotent and 0 is a reflexive vertex. Moreover, it satisfies the conditions of Case 1, so we are done.

Case 4. f is a minority operation on $\{0, n-2\}$.
Since $f(0, 0, n-2) = n-2$, we get $f(0, 0, x) = x$ by applying Lemma 4.4 to $f(x, x, y)$. So we have $f(0, 0, n-1) = n-1$. Similarly, $f(0, n-1, 0) = n-1$. Since $(n-1, n-1) \notin E(H)$, we get a contradiction.

Hence, $(Pol(\mathcal{H}^c))|_B$ consists of projections only and by Theorem 2.9 we get that CSP($\mathcal{H}^c$) is **NP**-complete, which by Proposition 2.5 allows us to conclude that RET(H) is **NP**-complete. □

5. Pseudotrees. Recall that a pseudotree is a connected graph containing at most one cycle. We will now prove the following theorem.

THEOREM 5.1. *Let H be a pseudotree. If $H|_{loop(H)}$ is disconnected or if H contains a cycle on $C \subseteq V(H)$ such that* RET($H|_C$) *is* **NP**-*complete, then* RET(H) *is* **NP**-*complete. Otherwise,* RET(H) *is in* **P**.

The proof is divided into two parts: in Section 5.1, we study a special type of pseudotrees that we call *balloons*, and we present the complete proof in Section 5.2.

5.1. Balloons. A *balloon H* is an irreflexive cycle with a pendant path such that the only vertex in H having a loop is the unique leaf, see Figure 5.1. The aim of this section is to prove the complexity of RET(H) for all balloons.

We denote the ballon having an irreflexive n-cycle ($n \geq 3$) with a pendant path of length m ($m \geq 1$), where only the leaf vertex is looped, by $B_{n,m}$. The vertices in the cycle are numbered $\{0, \ldots, n-1\}$, i ($1 < i < n-1$) is adjacent to $i+1$ and $i-1$, 0 is adjacent to $n-1$ and the m vertices in the path are numbered $\{n, \ldots, n+m-1\}$ where n is adjacent to 0, $n+m-1$ is looped, and $n+j$ is adjacent to $n+j+1$ ($0 \leq j \leq m-2$).

LEMMA 5.2. RET($B_{6,m}$) *is* **NP**-*complete for all* $m \geq 1$.

Proof. Consider the subalgebra $A = N(3) = \{2, 4\}$ of $B_{6,m}$. We show that $(Pol(B_{6,m}))|_A$ does not contain any max, min, majority, or minority operation.

Case 1: max operation. We assume that $f|_A$ is the max operation. Since $(5, 1)$ is a neighbour of both $(4, 2)$ and $(0, 0)$ we have $f(5, 1) \in N(4) \cap N(0) = \{5\}$. Similarly, $f(1, 3) = 3$ since $(1, 3)$ is a neighbour of both $(2, 4)$ and $(2, 2)$, so $f(1, 3) \in N(4) \cap N(2) = \{3\}$. With similar arguments we have $f(0, 2) = 2$, and using this result we get

$f(5,1) = 1$, since $(5,1)$ is a neighbour of both $(0,0)$ and $(0,2)$. This is a contradiction since we cannot have $f(5,1) = 5$ and $f(5,1) = 1$, so $(Pol(B_{6,m}))|_A$ does not contain max.

Case 2: min operation. Analogous to Case 1.

Case 3: majority operation. Assume that $f|_A$ is the majority operation. Since $(3,5,1)$ is a neighbour of both $(2,4,2)$ and $(4,4,2)$ we have $f(3,5,1) \in N(2) \cap N(4) = \{3\}$. Now, $(1,3,3)$ is a neighbour of both $(2,4,4)$ and $(2,2,2)$, so $f(1,3,3) \in N(4) \cap N(2) = \{3\}$. Using $f(1,3,3) = 3$ and analogous arguments to those above, we get $f(0,2,2) = 2$. Again repeating the argument and using $f(0,2,2) = 2$ we get $f(5,1,1) = 1$, and finally using $f(5,1,1) = 1$ we get that $f(4,0,0) = 0$. Now, since $(3,5,1)$ is a neighbour of $(4,0,0)$ we have a contradiction because $3 \notin N(0)$. Thus, $(Pol(B_{6,m}))|_A$ does not contain the majority function.

Case 4: minority operation. Analogous to the majority case. □

Now we present the complexity classification of $\text{Ret}(B_{n,m})$.

Lemma 5.3. *Let $B_{n,m}$ be a balloon. If $n = 4$, i.e., the length of the cycle is 4, then $\text{Ret}(B_{n,m})$ is in* **P** *and, otherwise, $\text{Ret}(B_{n,m})$ is* **NP***-complete.*

Proof. Assume that the cycle has length 4. Then, there exists two vertices a, b on the cycle satisfying $N(a) = N(b)$. By Lemma 2.3, $\text{Ret}(B_{4,m})$ and $\text{Ret}(B_{4,m} - a)$ are polynomial-time equivalent problems. Since $B_{4,m} - a$ is a path with a single loop, $\text{Ret}(B_{4,m} - a)$ and $\text{Ret}(B_{4,m})$ are in **P** [7].

As for hardness, we first note that Vikas [16] proved that $\text{Ret}(B_{3,1})$ is **NP**-complete. Moreover, we know from Lemma 5.2 that $\text{Ret}(B_{6,m})$ is **NP**-complete for all $m \geq 1$. We now show that $\text{Ret}(B_{n,m})$ is **NP**-complete in the remaining cases, i.e., when $n = 3$ and $m > 1$, $n = 5$, and $n \geq 7$. Given the graph $B_{n,m}$, construct the reflexive graph H such that $V(H) = V(B_{n,m})$ and $E(H)$ is defined by the following pp-formula:

$$E(H)(x,y) \equiv_{pp} \exists z E(x,z) \wedge E(z,y)$$

where E is the edge relation of $B_{n,m}$. Since H is pp-definable from $B_{n,m}$ it follows from Proposition 2.7 that $\text{Ret}(B_{n,m})$ is **NP**-complete if $\text{Ret}(H)$ is **NP**-complete. The graph H has different properties depending on whether n is even or odd (see Figures 5.2 and 5.3) so the proof is divided into two parts.

Case 1: n is even (see Figure 5.2). All vertices of H are looped, the even vertices in $\{0, \dots, n-1\}$ form a cycle ($2i$ is adjacent to $2(i+1)$ and $2(i-1)$), $n-2$ is adjacent to 0, and the only vertex adjacent to this cycle is $n+1$ which is adjacent to 0. Let k be the largest even number $\leq n/2$ and $j = \lfloor n/4 \rfloor$, then $H|_{N^j(k)}$ (i.e., the graph induced (in H) by the vertices in $N^j(k)$) is the reflexive $n/2$-cycle for which the retraction problem is **NP**-complete (remember that $n \geq 8$). Thus, by Proposition 2.8 we get that $\text{Ret}(H)$ is **NP**-complete.

Case 2: n is odd (see Figure 5.3). All vertices of H are looped, $2i$ is adjacent to $2(i-1)$ and $2(i+1)$ (where $0 < i < (n-1)/2$), $2j+1$ is adjacent to $2j-1$ and $2j+3$ (for $0 < j < (n-3)/2$), 0 is adjacent to $n-2$ and 1 is adjacent to $n-1$, so the vertices in $\{0, \dots, n-1\}$ form a reflexive cycle. Similarly for the vertices $j \in \{n+2, \dots, n+m-1\}$ we have that j is adjacent to $j+2$ and $j-2$, n is adjacent to 1 and $n-1$, $n+1$ is adjacent to 0, and $n+m-1$ is adjacent to $n+m-2$. Now consider the graph H' induced (in H) by the even vertices in $\{0, \dots, n-1\}$ together with all the vertices in $\{n, \dots, n+m-1\}$. It is easy to see that H' is the reflexive $\lceil n/2 \rceil + m$-cycle for which retraction is **NP**-complete (remember that $n \geq 5$, or $n = 3$ and $m \geq 2$). Now, the graph H retracts to H' by the retraction defined below.

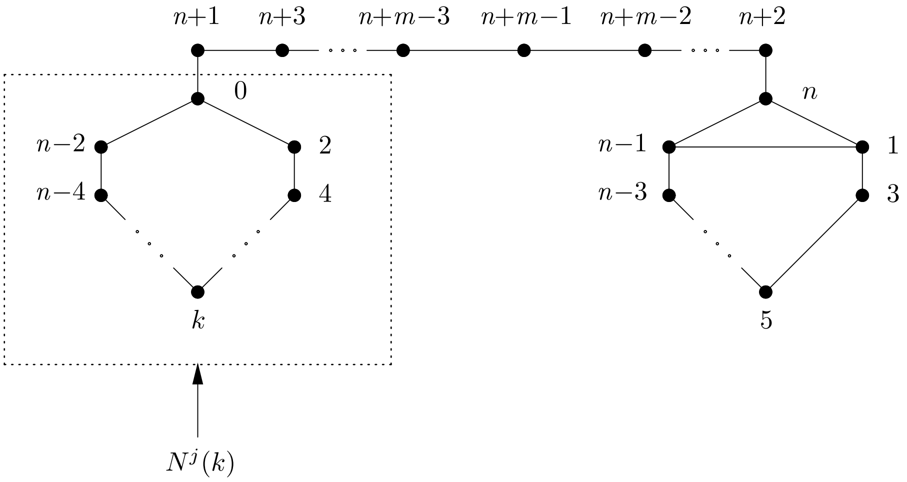

FIG. 5.2. *The reflexive graph H when n is even.*

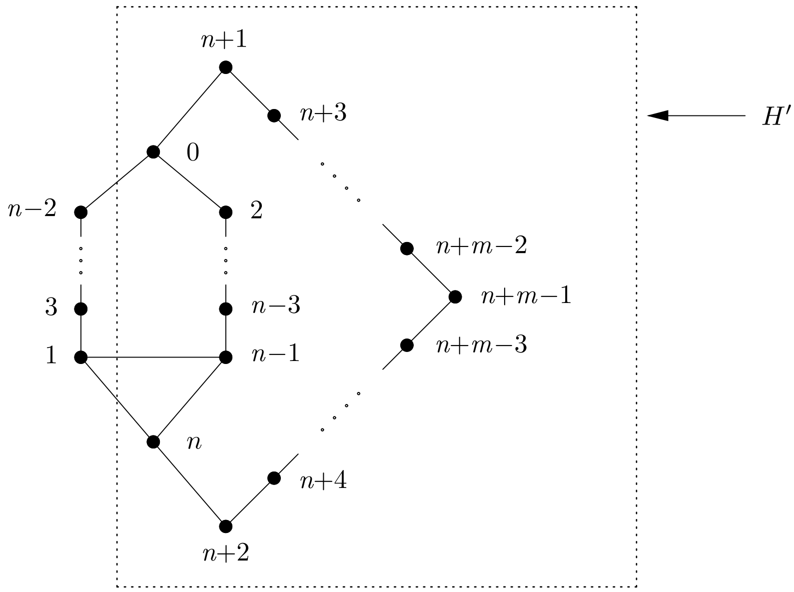

FIG. 5.3. *The reflexive graph H when n is odd.*

- $r(i) = i$ for all $i \in V(H')$,
- $r(i) = n - i$ for all $i \in V(H) \setminus V(H')$.

Hence, by Proposition 2.1 we get that $\textsc{Ret}(H)$ is **NP**-complete. □

5.2. Main result. A *leaf* in a graph is a vertex a having exactly one neighbour (not counting itself). We categorize leaves into four classes depending on loops in their neighborhoods: let a be a leaf and b its unique neighbour. If $bb \in E(H)$ and $aa \in E(H)$, we say that a is of type $(\circlearrowleft, \circlearrowleft)$; if $bb \notin E(H)$ but $aa \in E(H)$, then a is of type $(\cdot, \circlearrowleft)$, and the remaining two classes are defined analogously.

LEMMA 5.4. *Let H be a connected graph such that $|V(H)| \geq 3$ and $a \in V(H)$ is a leaf of type $(\circlearrowleft, \circlearrowleft)$, $(\circlearrowleft, \cdot)$, or $(\cdot, \cdot)$. Then, the problems* $\textsc{Ret}(H)$ *and* $\textsc{Ret}(H - a)$

are polynomial-time equivalent.

Proof. Let b be the unique neighbour of a and let c be a neighbour of b such that $a \neq c$. We consider three cases depending on the type of a: if a is of type $(\circlearrowleft, \circlearrowleft)$, then $N(a) = \{a, b\} \subseteq N(b)$ and the same holds if a is of type $(\circlearrowleft, \cdot)$. If a is of type $(\cdot, \cdot)$, then $N(a) = \{b\} \subseteq N(c)$. In all these cases, there is a vertex a' such that $N(a) \subseteq N(a')$. Now, it follows from Corollary 2.2 that there is a polynomial-time reduction from $\textsc{Ret}(H - a)$ to $\textsc{Ret}(H)$. For the reduction in the opposite direction, let G be an arbitrary instance of $\textsc{Ret}(H)$. As a consequence of $N_H(a) \subseteq N_H(a')$ we have that G retracts to H if and only if there is a retraction r from G to H such that a is the unique vertex in G that is mapped to a in H. Moreover, since a is a leaf in H the set $N_G(a) \setminus \{a\}$ must be mapped to the vertex $b = N_H(a) \setminus \{a\}$ by r. Denote by G'' the graph resulting from identifying the vertices in $N_G(a) \setminus \{a\}$ to b. It should be clear that G retracts to H if and only if G'' retracts to H. Since a is a leaf in G'' it is obvious that G'' retracts to H if and only if $G'' - a$ retracts to $H - a$, and the result follows. □

We can now prove the main theorem of this article.

Proof. (of Theorem 5.1) If $H|_{loop(H)}$ is disconnected, then $\textsc{Ret}(H)$ is **NP**-complete by Lemma 3.2 so we assume henceforth that $H|_{loop(H)}$ is connected.

Apply Lemma 5.4 to H repeatedly until it is not applicable anymore; let H' be the resulting graph. Note that $H'|_{loop(H')}$ is still connected and that $\textsc{Ret}(H)$ and $\textsc{Ret}(H')$ are polynomial-time equivalent problems. First, if H' is a tree, then $|V(H')| \leq 2$ (otherwise Lemma 5.4 can be applied) and $\textsc{Ret}(H')$ is in **P**. Hence, H' contains a (unique) cycle. If H' is a cycle, then the complexity of $\textsc{Ret}(H)$ follows from Corollary 4.2.

So we can assume that H' contains a cycle and at least one leaf. First of all, H' does not contain any leaves of type $(\circlearrowleft, \circlearrowleft)$, $(\circlearrowleft, \cdot)$, or $(\cdot, \cdot)$ by its construction. H' cannot contain two leaves of type $(\cdot, \circlearrowleft)$ since this would imply that $H'|_{loop(H')}$ is disconnected. Thus, H' contains exactly one leaf a (which is of type $(\cdot, \circlearrowleft)$). It also follows that a is the only vertex in H' with a loop: if the neighbour of a has a loop, then a is of type $(\circlearrowleft, \circlearrowleft)$, and if a non-neighbour has a loop, then $H'|_{loop(H')}$ is disconnected. In other words, H' is a balloon and the result follows from Lemma 5.3. □

By combining Proposition 2.4 and Theorem 5.1 we have the following complexity classification of $\textsc{Ret}(H)$ for every pseudoforest H.

- $\textsc{Ret}(H)$ is **NP**-complete when the looped vertices in a connected component of H induce a disconnected graph, H contains a cycle on at least 5 vertices, H contains a reflexive 4-cycle, or H contains an irreflexive 3-cycle.
- $\textsc{Ret}(H)$ is in **P** for all other pseudoforests H.

Acknowledgement. The authors would like to thank an anonymous referee for pointing out several errors in a previous version of the proof of Lemma 3.2.

REFERENCES

[1] L. Barto, M. Kozik, and T. Niven. The CSP dichotomy holds for digraphs with no sources and no sinks (a positive answer to a conjecture of Bang-Jensen and Hell). *SIAM Journal on Computing*, 38(5):1782–1802, 2009.

[2] A. Bulatov. H-coloring dichotomy revisited. *Theoretical Computer Science*, 349(1):31–39, 2005.

[3] A. Bulatov. A dichotomy theorem for constraint satisfaction problems on a 3-element set. *Journal of the ACM*, 53(1):66–120, 2006.

[4] A. Bulatov, P. Jeavons, and A. Krokhin. Classifying the computational complexity of constraints using finte algebras. *SIAM Journal on Computing*, 34(3):720–742, 2005.

[5] D. Cohen and P. Jeavons. *Handbook of Constraint Programming*, chapter 8: The complexity of constraint languages. Elsevier, 2006.

[6] T. Feder and P. Hell. List homomorphisms to reflexive graphs. *Journal of Combinatorial Theory, ser. B*, 72:236–250, 1998.

[7] T. Feder, P. Hell, and J. Huang. List homomorphisms and circular arc graphs. *Combinatorica*, 19:487–505, 1999.

[8] T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1998.

[9] P. Hell and J. Nešetřil. On the complexity of H-coloring. *Journal of Combinatorial Theory, ser. B*, 48:92–110, 1990.

[10] P. Hell and J. Nešetřil. *Graphs and Homomorphisms*. Oxford University Press, 2004.

[11] P. Jeavons, D. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44(4):527–548, 1997.

[12] R. Pöschel and L. Kalužnin. *Funktionen- und Relationenalgebren*. DVW, Berlin, 1979.

[13] E. Post. The two-valued iterative systems of mathematical logic. *Annals of Mathematical Studies*, 5:1–122, 1941.

[14] T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th ACM Symposium on Theory of Computing (STOC-78)*, pages 216–226, 1978.

[15] N. Vikas. Compaction, retraction, and constraint satisfaction. *SIAM Journal on Computing*, 33(4):761–782, 2004.

[16] N. Vikas. A complete and equal computational complexity classification of compaction and retraction to all graphs with at most four vertices and some general results. *Journal of Computer and System Sciences*, 71:406–439, 2005.