Dynamic Definability

Erich Grädel
Mathematische Grundlagen der Informatik,
RWTH Aachen University
graedel@logic.rwth-aachen.de

Sebastian Siebertz
Logik und Semantik, TU Berlin
sebastian.siebertz@tu-berlin.de

ABSTRACT

We investigate the logical resources required to maintain knowledge about a property of a finite structure that undergoes an ongoing series of local changes such as insertion or deletion of tuples to basic relations. Our framework is closely related to the *Dyn-FO*-framework of Patnaik and Immerman and the *FOIES*-framework of Dong, Libkin, Su and Wong, and also builds on work of Weber and Schwentick. We assume that the dynamic process starts with an arbitrary, nonempty structure, but in contrast to previous work, we assume that, in general, structures are unordered. We show how to modify known dynamic algorithms for symmetric reachability, bipartiteness, k-edge connectivity and more, to work also without an order and with dynamic processes starting at an arbitrary graph. A history independent dynamic system (also called deterministic or memoryless) is one that maintains all auxiliary information independent of the update order. In 1997, Dong and Su posed the problem whether there exist history independent dynamic systems with FO-updates for symmetric reachability or bipartiteness. We give a positive answer to this question. We further show that there is a history independent system for tree isomorphism with FO+C-updates. On the other hand we show that on unordered structures first-order logic is too weak to maintain enough information to answer the equal cardinality query and the tree isomorphism query dynamically.

Categories and Subject Descriptors

F.4.1 [**Mathematical Logic**]: Model Theory—*Finite Model Theory, Definability Theory*; F.1.m [**Models of Computation**]: Miscellaneous—*Dynamic Complexity, Incremental Computation*; H.2.4 [**Systems**]: Relational Databases

General Terms

Theory

ICDT 2012, March 26–30, 2012, Berlin, Germany.

Keywords

Finite Model Theory, Dynamic Complexity, DynFO, FOIES

1. INTRODUCTION

We are interested in some arbitrary but fixed query on a finite structure $\mathfrak{A}$ which is subject to an ongoing sequence of local changes, and after each change the answer to the query should remain available. As an example, consider a database application. A relational database is basically a finite relational structure $\mathfrak{A}$ and a fundamental query is the transitive closure of one of its relations. However, in reality a database is a dynamic object. Elements are constantly inserted in and deleted from the database and relations between elements change by insertion and deletion of tuples. Further, the query of interest usually does not need to be answered only once, but again and again. It thus lies at hand to maintain auxiliary relations that provide information about computations already performed on previous stages of $\mathfrak{A}$. In the context of databases, such relations are called materialized views. Other typical examples for applications where this dynamic view is appropriate are modeling of operating systems, embedded systems and many more.

Dynamic algorithms (i.e. algorithms maintaining dynamic data structures) for concrete problems have been studied for over three decades and many sophisticated techniques have been developed. A general theory of dynamic complexity providing structural results was first introduced by Miltersen et al. [19] and Patnaik and Immerman [20]. Motivated by the success of the descriptive approach to traditional complexity theory, Patnaik and Immerman [20] provided a descriptive approach to dynamic complexity theory. Their setting, called *Dyn-FO*, is designed to handle classes of structures, and auxiliary data consists of relations over the universe of the structures. In particular, no high level data structures are used and only a polynomial amount of auxiliary space is available. Weber and Schwentick [21] provide an elegant generalization to this setting which clarifies the role of precomputation. The initialization of the auxiliary structure and all updates are handled by logical interpretations. This allows for an analysis with well-known methods from finite model theory, and furthermore, all results obtained are readily usable for relational databases. A similar formalism for database applications, called *FOIES*, is defined and studied by Dong, Libkin, Su and Wong [8, 2, 4].

The complexity of a property in the descriptive context is measured by the power of logics required to express the property. Dynamic systems with low update costs, such as systems with updates expressed in first-order logic (FO) or

first-order logic with counting (FO+C) are of special interest. The expressive power of these logics corresponds exactly to the power of relational calculus or SQL on relational databases, and to the circuit complexity classes AC^0 or TC^0 on ordered structures [17, 16, 1]. Many properties that are not expressible in FO or FO+C in the classical static context can indeed be handled with FO or FO+C updates in the dynamic context. Examples for such properties are reachability in acyclic directed graphs [8], reachability in undirected graphs, k-edge connectivity and bipartiteness [20]. Structural results about dynamic classes have been provided for ordered structures, including suitable reduction concepts [20, 15, 21] and complete problems [15].

In this paper, we adapt the descriptive setting of Weber and Schwentick [21], i.e. we consider finite relational structures where auxiliary data consists of relations over the domain of the input structure. Initialization of the auxiliary structure and all updates are handled by logical interpretations. On ordered structures all considered logics correspond to well known complexity classes. In contrast to [20] and [21] we do, however, not assume that the given structures are ordered. It is a well known fact in finite model theory that the relationship between computational complexity and logical definability is much less tight on unordered finite structures compared to ordered ones. On ordered structures most of the major complexity classes can be precisely characterized by logics that either extend first-order by suitable operators such as transitive closure, fixed points etc., or by appropriate restrictions of, say, second-order logic. We refer to [11] for background on results of this kind and on the open problems concerning the relationship between logics and complexity classes.

On unordered structures, precise logical characterizations are known only for complexity classes such as NP and above. In particular it is major open problem whether there exist logics that capture Ptime or Logspace on the class of all finite structures. Thus, our focus on unordered finite structures means that our study is more about *dynamic definability* rather than *dynamic complexity*. There are many results on the power of dynamic systems on ordered structures, or equivalently on structures where the dynamic process starts with an empty structure (so that an ordering can be recovered from the sequence of insertions), but only few results on unordered structures when starting the dynamic process on structures with non-empty relations. It is our main interest to investigate whether these results also hold in the unordered setting. We show that many upper bounds for concrete problems indeed carry over to the unordered setting. Some dynamic systems from the literature directly translate into our setting. For instance the construction in [8] for reachability in acyclic directed graphs does not make use of an ordering at all. However, other known results on dynamic complexity crucially rely on the availability of an ordering such as the ones in [20, 7] concerning reachability in undirected graphs. We shall show how the undirected reachability query can be handled in the unordered setting. We shall establish a suitable reduction concept which is slightly more complicated than in the ordered setting. It will follow by a reduction from reachability that bipartiteness, k-vertex disjoint paths, k-edge connectivity and clique cover-2 are maintainable in the unordered setting as well.

In the setting of Dong and Su [6] the auxiliary relations for a structure $\mathfrak{A}$ are restricted to be independent of the order of the updates that lead to $\mathfrak{A}$. We call a dynamic system with this property history independent (in [6] it is called deterministic and in [20] memoryless). The advantage of history independent dynamic systems in the unordered setting is that, in contrast to non history independent systems, they can deal with any fixed number of updates at the same time and thus allow for a simpler reduction concept. It was posed as an open question in [6] whether there is a history independent dynamic system with FO-updates for reachability in undirected graphs or bipartiteness. We shall give a positive answer to that question. We shall further show how to modify the algorithm of Etessami [12] to handle tree isomorphism with FO+C-updates in a history independent fashion.

Providing lower bounds on the power of dynamic systems on ordered structures amounts to providing lower bounds for a general model of computation. It was shown by Etessami [12] that this is also the case when the dynamic process starts with an empty structure, as order and arithmetic can be built incrementally. Thus proving lower bounds in the ordered setting is probably beyond reach. In [7, 9] a strict hierarchy based on the arity of allowed auxiliary relations is established, but no lower bounds on systems with auxiliary relations of arbitrary arity are known. The situation is different when the dynamic process starts with a non-empty unordered structure. We shall show that in our unordered setting neither equal cardinality nor tree isomorphism can be handled with FO-updates. Our method is similar to that of [7] using Ehrenfeucht-Fraïssé games. The unmaintainability of equal cardinality induces a separation of the dynamic complexity class $\mathrm{Dyn}(\mathsf{FO+C}, \mathsf{FO})$ from $\mathrm{Dyn}(\mathsf{FO+C}, \mathsf{FO+C})$. The separation on ordered structures, i.e. the separation of classes with AC^0-updates from classes with TC^0-updates, is a major open problem in dynamic complexity theory. Our result implies that there is no history independent *FOIES* for equal cardinality and tree isomorphism. The former was already shown in [6], the latter is the first result of nonexistence of a history independent *FOIES* for a query over non-unary input vocabulary. We also show that the auxiliary relations used by our dynamic system to maintain the transitive closure of an undirected graph do not suffice to maintain the transitive closure of a directed graph.

2. DYNAMIC DEFINABILITY CLASSES

In this section we fix our notation and define dynamic problems, dynamic definability classes and dynamic reductions. We give the formal definitions along the lines of an example, the dynamic symmetric reachability problem.

All considered structures are finite over finite vocabularies that contain only relation and constant symbols. Let $\tau = \{R_1, \ldots, R_k, c_1, \ldots, c_s\}$ be a signature. Denote by $\mathrm{Fin}(\tau)$ all finite τ-structures and by $\mathrm{Fin}_n(\tau)$ all τ-structures with n elements. For a τ-structure $\mathfrak{A}$ over universe A we write $\mathfrak{A} = (A, R_1^{\mathfrak{A}}, \ldots, R_k^{\mathfrak{A}}, c_1^{\mathfrak{A}}, \ldots, c_s^{\mathfrak{A}})$. For simplicity we assume that the universe A of a structure $\mathfrak{A} \in \mathrm{Fin}(\tau)$ is $[n] := \{0, \ldots, n-1\}$. A decision problem on τ-structures is a Boolean query on τ-structures, i.e. a class $\mathcal{S}$ of τ-structures, such that if $\mathfrak{A} \cong \mathfrak{B}$ then $\mathfrak{A} \in \mathcal{S} \Leftrightarrow \mathfrak{B} \in \mathcal{S}$. A query $\mathcal{S}$ is *definable* in a logic $\mathcal{L}$ if there is a sentence φ of $\mathcal{L}(\tau)$ such that for every $\mathfrak{A} \in \mathrm{Fin}(\tau)$ we have $\mathfrak{A} \in \mathcal{S} \Leftrightarrow \mathfrak{A} \models \varphi$. If $\varphi(x_1, \ldots, x_k)$ is a formula with k free variables we write $\varphi^{\mathfrak{A}}$ for the k-ary relation $\{(a_1, \ldots, a_k) \in A^k : \mathfrak{A} \models \varphi(a_1, \ldots, a_k)\}$. We focus mostly on first-order logic FO and its well known ex-

tensions with transitive closure operators FO+TC or inflationary fixed-point operators IFP, as well as their counting extensions FO+C, FO+TC+C and IFP+C. We assume familiarity with these logics, modern treatments of the above topics can be found in [11, 10, 18]. We write $\overline{x}$ for a finite sequence $(x_1, \ldots, x_k)$ and usually leave it to the context to determine the length of a sequence.

Each decision problem $\mathcal{S}$ together with a set of operations induces the dynamic problem of whether a sequence of operations applied to a structure $\mathfrak{A}$ produces a structure with the desired property $\mathcal{S}$. Thus, the elements of a dynamic problem $\mathcal{D}$ are pairs $(\mathfrak{A}, w)$, consisting of a structure $\mathfrak{A}$ and a sequence w of operations, such that the structure resulting from $\mathfrak{A}$ after applying w is in $\mathcal{S}$. For each vocabulary $\tau = \{R_1, \ldots, R_k, c_1, \ldots, c_s\}$ we consider a set $\Sigma_{can}(\tau)$ of operation symbols, together with an associated set $\Delta_{can}(\tau)$ of operations. $\Sigma_{can}(\tau)$ contains the symbols

- Insert_R for each relation symbol $R \in \tau$ with associated operations $\text{Insert}_R(\overline{a}) \in \Delta_{can}(\tau)$, where $\overline{a} \in \mathbb{N}^k$ for $k = \text{arity}(R)$,
- Delete_R for each relation symbol $R \in \tau$ with associated operations $\text{Delete}_R(\overline{a}) \in \Delta_{can}(\tau)$, where $\overline{a} \in \mathbb{N}^k$ for $k = \text{arity}(R)$ and
- Set_c for each constant symbol $c \in \tau$ with associated operations $\text{Set}_c(a) \in \Delta_{can}(\tau)$ for $a \in \mathbb{N}$

with the following semantics. If any of the parameters of an operation is not in the universe of the structure, the operation has no effect. Otherwise, an insert or delete operation adds the given tuple to the corresponding relation or deletes the given tuple, respectively. A set operation sets the corresponding constant to be the given element. The operations are called the canonical operations for τ. On undirected graphs we define the semantics of the canonical operations $\text{Insert}_E(a, b)$ ($\text{Delete}_E(a, b)$) such that they insert (delete) both the edge (a, b) and (b, a) to (from) the given graph.

For a sequence $w = \delta_1 \ldots \delta_m \in \Delta^*_{can}(\tau)$ of operations and a structure $\mathfrak{A}$, we define $w(\mathfrak{A})$ to be the result of subsequently applying the operations to $\mathfrak{A}$, and $\mathfrak{A}$ if $w = \epsilon$. For a class $\mathcal{C}$ of structures denote by $w(\mathcal{C})$ the class $\{w(\mathfrak{A}) : \mathfrak{A} \in \mathcal{C}\}$.

DEFINITION 2.1. *Let $\mathcal{S}$ be a Boolean query on τ-structures. The dynamic problem $\mathcal{D}(\mathcal{S})$ associated with $\mathcal{S}$ is the set of pairs $D = (\mathfrak{A}, w)$ where $\mathfrak{A} \in Fin(\tau)$ and $w \in \Delta^*_{can}(\tau)$ is an update sequence with $w(\mathfrak{A}) \in \mathcal{S}$. The query $\mathcal{S}$ is called the underlying static problem of $\mathcal{D}(\mathcal{S})$.*

We want to handle dynamic problems by incremental evaluation systems. These systems allow auxiliary relations over the universe of the input structure $\mathfrak{A}$. Thus, we are limited to a polynomial amount of auxiliary space and in particular, no high level data structures are used. Given $\mathfrak{A}$, the auxiliary relations are initialized by a logical interpretation in an expressive logic such as IFP or IFP+C. Each update should be handled fast, making use of the auxiliary data. In our context this means the updates are handled by logical interpretations in a simple logic, such as FO or FO+C.

Example: Symmetric Reachability. The static reachability problem on undirected graphs, SymmReach, consists of all undirected graphs $\mathcal{G}$ with distinguished vertices s and t such that t is reachable in $\mathcal{G}$ from s. The canonical operations are the insertion and deletion of undirected edges as well as setting of s and t to specific vertices of the graph. For an operation sequence w call $w(\mathcal{G})$ the graph that results from applying all insert and delete operations from w to $\mathcal{G}$ and call $w(s)$ and $w(t)$ the constants that result from applying the corresponding set operations to s and t. A pair $(\mathfrak{G}, w)$, is in $\mathcal{D}(\text{SymmReach})$ if $w(t)$ is reachable from $w(s)$ in $w(\mathcal{G})$. We want to stress again that contrary to previous results from [7, 20] we assume no order on the nodes and an initialization by purely logical means. This example is meant to show that difficulties arise when no order is present. We define an incremental evaluation system for SymmReach that initializes and maintains two ternary relations F and R with the following properties. After the initialization we have $Fxyz$ if and only if (y, z) is an edge in $\mathcal{G}$ and lies on a shortest path from x to some other vertex. For each vertex x, $Fxyz$ induces an acyclic directed graph F_x. At all times during the update process it will hold for each vertex x that F_x is acyclic and connected, and a vertex z is reachable from x if and only if $x = z$ or there exists an edge (y, z) with $(y, z) \in F_x$. Also it holds at all times that $Rxyz$ expresses that in F_x vertex z is reachable from vertex y. F and R can be initialized by formulae in inflationary fixed point logic.

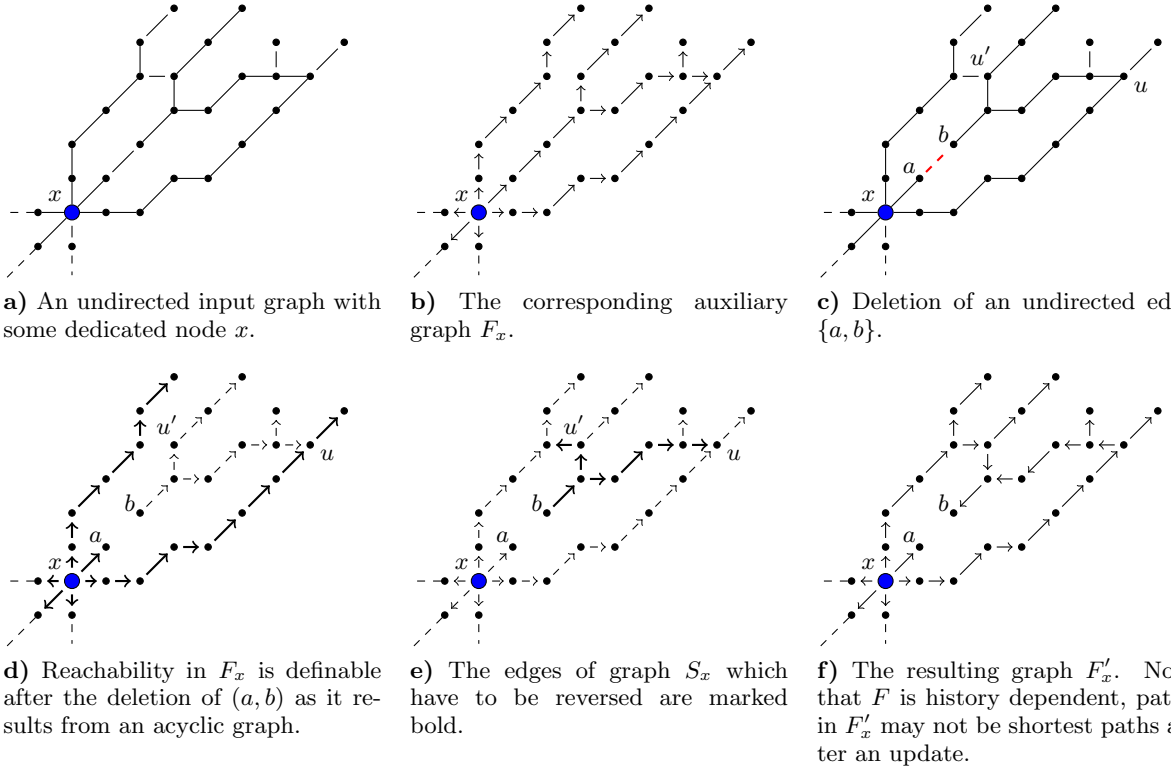
$$Fxyz := [\mathsf{ifp}\, Guv.(u = x \wedge Euv) \vee \exists t(Gtu \wedge (Euv \wedge \neg \exists s Gsv \wedge x \neq v)]yz.$$

$$Rxyz := [\mathsf{ifp}\, Huv.(u = v \vee Fxuv \vee \exists w(Huw \wedge Hwv))]yz.$$

Stage 1 of the fixed-point iteration for F contains the edges that lead to direct successors of vertex x, $F_1 = \{(x, v) \in E\}$. Stage F_{i+1} additionally contains those edges that lead from F_i to a vertex which which was not reachable so far. That is, each stage F_i is the set $\{(u, v) \in E :$ there is a shortest path from x to v in $\mathcal{G}$ of length $\leq i$ which goes along edge $(u, v)\}$.

We now want to update the predicates F and R by first-order formulae whenever an update to $\mathcal{G}$ occurred. We have to agree on how updates are presented to the incremental evaluation system. Either we assume that an update is explicitly given, e.g. presented in the form $\text{Insert}_E(a, b)$. In this case the update formula for an insert has two free variables which are interpreted as a, b when the update occurs. Or we assume that the incremental evaluation system gets only the information that an insert operation occurred, i.e. that the edge relation E was updated. In this case we have to maintain an additional relation E^{old} which is always interpreted as the edge relation as it was before the update and changes are existentially quantified. We assume the second possibility, as we want to stress the unordered nature of an update, e.g. when an undirected edge $\{a, b\}$ is inserted to $\mathcal{G}$, we do not want to use the implicit order induced by the presentation of the update, but really treat the edge as undirected. The first approach is mostly used in the *Dyn-FO* framework by [20] and [21], whereas the second approach was used for example in [7].

Handling insert operations is rather simple. Let $\{a, b\}$ be the undirected edge that is inserted by the update. If neither a nor b are reachable from x, no edge is inserted into F_x, as F_x is supposed to remain connected. If both a and b are reachable, again no edge is added to F_x as no new reachability information is introduced. Thus F_x remains



a) An undirected input graph with some dedicated node x.

b) The corresponding auxiliary graph F_x.

c) Deletion of an undirected edge $\{a, b\}$.

d) Reachability in F_x is definable after the deletion of (a, b) as it results from an acyclic graph.

e) The edges of graph S_x which have to be reversed are marked bold.

f) The resulting graph F'_x. Note that F is history dependent, paths in F'_x may not be shortest paths after an update.

Figure 1: Visualization of the initialization and deletion step for symmetric reachability

acyclic. If only one of a or b is reachable, (a, b) or (b, a) is inserted, respectively. Assume that (a, b) is inserted (a is reachable from x and b is not reachable from x). Then the vertices of F_x and F_b form disjoint sets and the edges of F_b are added to F_x. Observe that F_x remains cyclefree and connected, and y is reachable from x if and only if there is a path from x to y in F_x. We obtain the newly updated predicates F' and R' as follows. Note that the predicate F is history dependent.

$$\begin{aligned} F'xyz :=& Fxyz \vee \exists a \exists b((a \neq b \wedge Eab \wedge \neg E^{old}ab) \wedge \\ & (Rxxa \wedge \neg Rxxb \wedge ((y = a \wedge z = b) \vee Fbyz))). \end{aligned}$$

$$\begin{aligned} R'xyz :=& Rxyz \vee \exists a \exists b((a \neq b \wedge Eab \wedge \neg E^{old}ab) \wedge \\ & (Rxxa \wedge \neg Rxxb \wedge ((Rxya \wedge Rbbz) \vee Rbyz))). \end{aligned}$$

Handling delete operations is more complicated. Let $\{a, b\}$ be the edge deleted by the update. To maintain the predicates with the desired properties we make use of a result of [8]. In acyclic directed graphs, we can maintain the transitive closure of the graph with no additional auxiliary relations but the transitive closure itself. Let

$$\begin{aligned} Pxyz \equiv& Rxyz \wedge \\ & [\neg(\exists a \exists b(a \neq b \wedge E^{old}ab \wedge \neg Eab \wedge Fxab)) \vee \\ & (\exists a \exists b((a \neq b \wedge E^{old}ab \wedge \neg Eab \wedge Fxab) \wedge \\ & \exists u \exists v(Rxyu \wedge Rxua \wedge Fxuv \wedge \neg Rxva \wedge \\ & Rxvz \wedge (v \neq b \vee u \neq a))))]. \end{aligned}$$

$Pxyz$ expresses reachability in F_x after the deletion of an edge. If neither (a, b) nor (b, a) exists in F_x then reachability does not change. If on the other hand one of the edges (a, b) or (b, a) exists in F_x then $Pxyz$ expresses reachability in F_x after the deletion of (a, b) or (b, a), respectively. See [8] for a proof that the above formula correctly updates the transitive closure in the acyclic graph F_x after the deletion of an edge.

We now update the predicate F as follows. All edges (y, z) from F_x with $Pxxy$ and $(y, z) \neq (a, b)$ remain in F_x. Furthermore it holds that every vertex which was reachable in F_x before the deletion of (a, b) is also reachable after the deletion (via a different path) if the following holds. From b some vertex u is reachable in F_x before the deletion of (a, b), which is either reachable also after the deletion of (a, b), or connected to a vertex v reachable after the deletion of (a, b) via an edge $\{u, v\}$ which is not represented by an edge in F_x. In this case the path from b to u in F_x can be reversed and then leads from u to b. Possibly the edge (v, u) has to be inserted. We define a graph S_x containing all edges that have to be reversed, and the edge (u, v) if it is needed, by the formula

$$\begin{aligned} Sxyz \equiv& \exists a \exists b((a \neq b \wedge E^{old}ab \wedge \neg Eab \wedge Fxab) \wedge \\ & [(Fxyz \wedge Pxby \wedge \neg Pxxy \wedge \exists u \exists v(\neg Pxxu \wedge \\ & Pxyu \wedge Pxxv \wedge Euv)) \vee (\neg Fxyz \wedge \neg Fxzy \wedge \\ & \neg Pxxy \wedge Pxxz \wedge Pxby \wedge Eyz)]). \end{aligned}$$

The edges of S_x are marked bold in Figure 1e). The edges of S_x form an acyclic graph and adding the reversed edges of S_x to F_x does not introduce cycles in F_x. There may be paths of F_x attached to S_x that do not lead to reachable vertices. These are accessed in another way if S_x is nonempty and

they have to be deleted if S_x is empty.

$$\begin{aligned}F'xyz :=&(\neg(\exists a\exists b(a \neq b \wedge E^{old}ab \wedge \neg Eab)) \wedge Fxyz)\vee\\ &\exists a\exists b(a \neq b \wedge E^{old}ab \wedge \neg Eab\wedge\\ &\quad[(Fxyz \wedge \neg Fxab \wedge \neg Fxba)\vee\\ &\quad(Fxyz \wedge Pxxy \wedge Pxxz \wedge (y,z) \neq (a,b))\vee\\ &\quad(Fxzy \wedge Sxyz)\vee\\ &\quad(Fxyz \wedge \neg Pxxy \wedge \neg Pxxz \wedge \exists u\exists v(Sxuv)\wedge\\ &\qquad\neg Sxyz)]).\end{aligned}$$

$$\begin{aligned}R'xyz :=&(\neg(\exists a\exists b(a \neq b \wedge E^{old}ab \wedge \neg Eab)) \wedge Rxyz)\vee\\ &\exists a\exists b(a \neq b \wedge E^{old}ab \wedge \neg Eab\wedge\\ &\quad[(Rxyz \wedge \neg Fxab \wedge \neg Fxba)\vee\\ &\quad(Fxab \wedge Pxyz \wedge (y,z) \neq (a,b))\vee\\ &\quad(Fxab \wedge \exists u\exists v(Pxxv \wedge Sxuv\wedge\\ &\qquad((Pxyv \wedge Pxzu) \vee (Pxyu \wedge Pxzy)\vee\\ &\qquad(Pxyv \wedge \exists w(Pxwu\wedge\\ &\qquad\quad\exists t(Fxwt \wedge \neg Sxwt \wedge Pxtz)))\vee\\ &\qquad(Pxyu \wedge \exists w(Pxwy\wedge\\ &\qquad\quad\exists t(Fxwt \wedge \neg Sxwt \wedge Pxtz))))))]).\end{aligned}$$

Finally we can query whether t is reachable from s by the FO-formula $Rsst$.

Interpretations. For symmetric reachability we used IFP-formulae to initialize several auxiliary relations over the universe of the structure. Updates were handled by a set of FO-formulae and finally the query answer was obtained by a single FO-formula. To group the initial formulae and update formulae we use the concept of logical interpretations.

DEFINITION 2.2. *Let $\mathcal{L}$ be a logic and σ,τ relational vocabularies. A k-dimensional $\mathcal{L}(\sigma,\tau)$-interpretation is given by a sequence $\mathcal{J}$ of $\mathcal{L}(\sigma)$-formulae consisting of a domain formula $\delta(\overline{x})$, for each relation symbol $R \in \tau$ (of arity r) a formula $\varphi_R(\overline{x}_1,\ldots,\overline{x}_r)$, and for each constant symbol $c \in \tau$ a formula $\varphi_c(\overline{x})$, where all tuples $\overline{x},\overline{x}_i$ have length k. All formulae may involve parameters. An $\mathcal{L}(\sigma,\tau)$-interpretation induces a mapping from σ-structures to τ-structures. For a τ-structure $\mathfrak{B}$ and a σ-structure $\mathfrak{A}$, we say that $\mathcal{J}$ interprets $\mathfrak{B}$ in $\mathfrak{A}$, in short $\mathcal{J}(\mathfrak{A}) = \mathfrak{B}$, if there exists a bijective map $h : \delta^{\mathfrak{A}} \to A$, called the coordinate map, such that*

- *for every relation $R^{\mathfrak{B}}$ of $\mathfrak{B}$ and all $\overline{a}_1,\ldots,\overline{a}_r \in \delta^{\mathfrak{A}}$*

$$\mathfrak{A} \models \varphi_R(\overline{a}_1,\ldots,\overline{a}_r) \Leftrightarrow (h(\overline{a}_1),\ldots,h(\overline{a}_r)) \in R^{\mathfrak{B}},$$

  i.e. $h^{-1}(R^{\mathfrak{B}}) = (\delta^{\mathfrak{A}})^r \cap \varphi_R^{\mathfrak{A}}$ and
- *for every constant $c^{\mathfrak{B}}$ of $\mathfrak{B}$ and all $\overline{a} \in \delta^{\mathfrak{A}}$*

$$\mathfrak{A} \models \varphi_c(\overline{a}) \Leftrightarrow h(\overline{a}) = c^{\mathfrak{B}},$$

  i.e. $h^{-1}(c^{\mathfrak{B}}) = \delta^{\mathfrak{A}} \cap \varphi_c^{\mathfrak{A}}$.

We call an interpretation simple *if it is one-dimensional and $\delta(x)$ is true for all x. We say that an interpretation has constant domain on a class $\mathcal{D}$ of structures if $\delta^{\mathfrak{B}_1} = \delta^{\mathfrak{B}_2}$ for all $\mathfrak{B}_1,\mathfrak{B}_2 \in \mathcal{D}$ which have the same universe.*

Each $\mathcal{L}(\sigma,\tau)$-interpretation induces a mapping from $\mathcal{L}(\tau)$-formulae to $\mathcal{L}(\sigma)$ formulae. For a τ-formula φ we write $\mathcal{J}(\varphi)$ for the interpreted formula. It holds that that $\mathcal{J}(\mathfrak{A}) \models \varphi \Leftrightarrow \mathfrak{A} \models \mathcal{J}(\varphi)$.

Example: Bipartiteness. We use a logical interpretation to reduce the bipartiteness problem in undirected connected graphs to the connectivity problem in undirected graphs. More precisely, we give a first-order interpretation $\mathcal{J}$ with parameters such that for each undirected connected graph $\mathcal{G} = (V,E)$ it holds that $\mathcal{G}$ is bipartite if and only if $\mathcal{J}(\mathcal{G},a,b)$ is not connected for all distinct $a,b \in V$. The interpretation is two-dimensional, i.e. not simple, but has constant domain.

Let $\mathcal{G} = (V,E)$ be an undirected connected graph. For each node $v \in V$ create two copies v_a and v_b. For each edge $\{u,v\} \in E$ we introduce two edges $\{u_a,v_b\}$ and $\{u_b,v_a\}$. Then $\mathcal{J}(\mathcal{G})$ is not connected if and only if $\mathcal{G}$ is bipartite. This is due to the fact that a graph is bipartite if and only if it contains no odd cycles. Formally the interpretation is defined by

- $\delta(x_1,y_1,x_2,y_2,a,b) := (x_1 = a \vee x_1 = b) \wedge (x_2 = a \vee x_2 = b) \wedge y_1 = y_1 \wedge y_2 = y_2$,
- $\varphi_E(x_1,y_1,x_2,y_2,a,b) := (x_1 = a \wedge x_2 = b \wedge Ey_1y_2) \vee (x_1 = b \wedge x_2 = a \wedge Ey_1y_2)$.

Incremental Evaluation Systems. An *Incremental Evaluation System (IES)* for a dynamic problem $\mathcal{D}(\mathcal{S})$ consists of a set of logical interpretations and an additional logical sentence φ. Given an initial structure $\mathfrak{A}$, the IES defines auxiliary relations over the universe of $\mathfrak{A}$ by an interpretation called the initial interpretation. Formally, it defines a structure $\mathfrak{B}$ of vocabulary $\sigma = \tau \dot\cup \tau'$, where τ' is the auxiliary vocabulary. After the initialization, each update in the operation sequence changes $\mathfrak{B} \restriction \tau$ as defined by the operation. Internally, it induces a change of $\mathfrak{B} \restriction \tau'$ by a corresponding logical interpretation. Thus the IES defines a mapping from τ-structures to σ-structures. This mapping should be a many-one reduction from $\mathcal{S}$ to the class $\mathcal{S}'$ of σ-structures which is defined by the given sentence φ, i.e. $\mathfrak{A} \in \mathcal{S} \Leftrightarrow \mathfrak{B} \in \mathcal{S}' \Leftrightarrow \mathfrak{B} \models \varphi$.

Given a τ structure $\mathfrak{A}$ and a τ'-structure $\mathfrak{A}'$ with the same $\tau \cap \tau'$-reduct (and thus in particular with the same universe) we write $\mathfrak{A} \sqcup \mathfrak{B}$ for their joint expansion to a $(\tau \cup \tau')$-structure.

DEFINITION 2.3. *Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be logics. Dyn$(\mathcal{L}_1,\mathcal{L}_2)$ is the class of all dynamic problems $\mathcal{D}(\mathcal{S})$, where $\mathcal{S}$ is a Boolean query on τ-structures, for which there exist*

- *an $\mathcal{L}_2$-definable class $\mathcal{S}'$ of $\tau \cup \tau'$-structures where τ' is disjoint from τ,*
- *a simple $\mathcal{L}_1(\tau,\tau\cup\tau')$-interpretation $\mathcal{J}$ (the initial interpretation),*
- *simple $\mathcal{L}_2(\tau\cup\tau',\tau\cup\tau')$-interpretations $\mathcal{J}_R^{ins}$, $\mathcal{J}_R^{del}$, $\mathcal{J}_c^{set}$ for each operation symbol Insert$_R$, Delete$_R$ and Set$_c$, respectively, (the update interpretations)*

such that for all sequences w of operations of $\mathcal{D}(\mathcal{S})$ the following hold:

If $|w| = 0$ then $\mathfrak{B}_0 := \mathcal{J}(\mathfrak{A}) \in \mathcal{S}' \Leftrightarrow \mathfrak{A} \in \mathcal{S}$.

If $|w| = i+1$, $w = v\delta$ where δ is an update with corresponding update interpretation $\mathcal{J}_\delta$ and $\mathfrak{B}_i$ is the result of the dynamic process of applying v to $\mathfrak{A}$, then

$$\mathfrak{B}_{i+1} := \mathcal{J}_\delta(w(\mathfrak{A}) \sqcup (\mathfrak{B}_i \restriction \tau')) \in \mathcal{S}' \Leftrightarrow w(\mathfrak{A}) \in \mathcal{S}.$$

Such a system of interpretations is called an $(\mathcal{L}_1,\mathcal{L}_2)$-IES for $\mathcal{D}(\mathcal{S})$.

To stress the difference between the power of logics on ordered and unordered structures we use notions from complexity theory whenever we are in the context of ordered structures. For example we say that equal cardinality is in $\mathrm{Dyn}(\mathsf{TC}^0, \mathsf{AC}^0)$ to state that the equal cardinality query allows for IES with an $\mathsf{FO}{+}\mathsf{C}$ initialization and FO updates on ordered structures. On unordered structures we will show that equal cardinality $\notin \mathrm{Dyn}(\mathcal{L}, \mathsf{FO})$ for any logic $\mathcal{L}$.

We now discuss dynamic reductions which allow us to show containment of many problems in a particular dynamic class without constructing IES from scratch. A dynamic reduction has the property that if a dynamic problem $\mathcal{D}_2$ has an $(\mathcal{L}_1, \mathcal{L}_2)$-IES and $\mathcal{D}_1$ is reducible to $\mathcal{D}_2$ via a suitable reduction, then $\mathcal{D}_1$ has an $(\mathcal{L}_1, \mathcal{L}_2)$-IES as well. The main idea of a dynamic reduction is the following. Most natural logics that we consider are closed under composition. Thus, if a change to an instance $\mathfrak{A}_1$ of $\mathcal{D}_1$ induces a bounded number of changes to an instance $\mathfrak{A}_2$ of $\mathcal{D}_2$, the update formulae for $\mathcal{D}_2$ can be composed to obtain a single update formula for $\mathcal{D}_1$. We present the details.

Let $\mathcal{D}_1 = \mathcal{D}(\mathcal{S}_1)$ and $\mathcal{D}_2 = \mathcal{D}(\mathcal{S}_2)$ be dynamic problems such that $\mathcal{S}_1$ is $\mathcal{L}_1$-reducible to $\mathcal{S}_2$ via interpretation $\mathcal{J}_r$. Let $\mathcal{D}_2 \in \mathrm{Dyn}(\mathcal{L}_1, \mathcal{L}_2)$ with an $(\mathcal{L}_1, \mathcal{L}_2)$-IES with initial interpretation $\mathcal{J}_i$. On input $\mathfrak{A}_1$ for $\mathcal{D}_1$ we can first create a corresponding instance $\mathfrak{A}_2 := \mathcal{J}_r(\mathfrak{A}_1)$ of $\mathcal{D}_2$. Assume $\mathcal{L}_1$ is closed under $\mathcal{L}_1$-interpretations. Then we can combine the interpretations $\mathcal{J}_i$ and $\mathcal{J}_r$ to define all required auxiliary structures to maintain property $\mathcal{S}_2$ on $\mathfrak{A}_2$.

Each (τ, τ')-interpretation $\mathcal{J}$ with constant domain induces a mapping $h_{\mathcal{J}}$ from $\mathrm{Fin}(\tau) \times \Delta_{can}(\tau)$ to a set of operations from $\Delta_{can}(\tau')$ by

$$\begin{aligned} h_{\mathcal{J}}(\mathfrak{A}, \delta) = &\{\mathrm{Insert}_R(\overline{a}) : R \in \tau', \mathcal{J}(\mathfrak{A}) \not\models R\overline{a}, \mathcal{J}(\delta(\mathfrak{A})) \models R\overline{a}\} \\ &\cup \{\mathrm{Delete}_R(\overline{a}) : R \in \tau', \mathcal{J}(\mathfrak{A}) \models R\overline{a}, \mathcal{J}(\delta(\mathfrak{A})) \not\models R\overline{a}\} \\ &\cup \{\mathrm{Set}_c(a) : \mathcal{J}(\mathfrak{A}) \not\models c = a, \mathcal{J}(\delta(\mathfrak{A})) \models c = a\}. \end{aligned}$$

We call the elements of $h_{\mathcal{J}}(\mathfrak{A}, \delta)$ the updates induced by δ on $\mathfrak{A}$ under interpretation $\mathcal{J}$. We say that $\mathcal{J}$ has the *bounded expansion property* if $|h_{\mathcal{J}}(\mathfrak{A}, \delta)| < k$ for some fixed $k \in \mathbb{N}$ and each $\mathfrak{A} \in \mathrm{Fin}(\tau)$ and each $\delta \in \Delta_{can}(\tau)$.

Given an interpretation with the bounded expansion property, each update to $\mathfrak{A}_1$ induces only a constant number of updates to $\mathfrak{A}_2$. All induced changes can thus be existentially quantified in a single formula. A difficulty arises in the unordered setting which is caused by the way in which the induced updates are accessed. We demonstrate this difficulty by an example.

Example 2.4. *The dynamic algorithm of Patnaik and Immerman [20] for symmetric transitive closure maintains a spanning forest of the given graph as auxiliary relation. This auxiliary relation depends on the update history. Assume that a problem $\mathcal{D}_1$ reduces to symmetric transitive closure and the insertion of a tuple induces the insertion of a bounded number of edges. The induced updates are obtained via existential quantification and are handled by a first-order formula which looks similar to the following: $\exists x_1 \exists y_1 (Exy_1 \wedge \neg E^{old} x_1 y_1 \wedge (\varphi_{update}(x_1, y_1) \wedge \exists x_2 \exists y_2 (\ldots)))$. By the nature of logics, when evaluating this formula, the updates are performed in parallel in all possible orders. The resulting auxiliary structure is thus the union of all auxiliary structures one obtains when performing those updates in some order. As a consequence, the forest structure of the auxiliary relation is possibly not maintained.*

We observe that in general the updates $h_{\mathcal{J}}(\mathfrak{A}_1, \delta)$ to $\mathfrak{A}_2$ induced by an update δ to $\mathfrak{A}_1$ obtained by purely logical means form a set, and not a sequence of updates. However, the auxiliary structure built in the IES for $\mathcal{D}_2$ may depend on the order in which the updates occur. If no order on the updates is definable there is no way to update the auxiliary relations correctly. We therefore further require that an order should be definable on the induced updates.

Let $\mathcal{J}$ be an $\mathcal{L}_2$-interpretation with the bounded expansion property inducing the mapping $h_{\mathcal{J}}$. Assume that furthermore there is an $\mathcal{L}_2$-formula $\varphi_<$ defining for each $(\mathfrak{A}, \delta)$ a partial order on A which is linear restricted to those elements actually appearing in $h_{\mathcal{J}}(\mathfrak{A}, \delta)$. Let $(\mathfrak{A}, w)$ be an instance of $\mathcal{D}_1$. We write $h^<_{\mathcal{J}}(w)$ for the sequence of updates that is obtained in the natural way by bringing the induced updates in the lexicographical order induced by $<$. Note that $h^<_{\mathcal{J}}(w)$ may depend on $\mathfrak{A}$.

We call a tuple $(\mathcal{J}_r, \mathcal{J}, \varphi_<)$ *suitable* for $\mathcal{D}_1$ and $\mathcal{D}_2$ if for all $\mathfrak{A}$

$$(\mathfrak{A}, w) \in \mathcal{D}_1 \Leftrightarrow (\mathcal{J}_r(\mathfrak{A}), h^<_{\mathcal{J}}(w)) \in \mathcal{D}_2.$$

In this case, if $\mathcal{L}_2$ is closed under $\mathcal{L}_2$-interpretation, we can react to each update of $\mathfrak{A}$ with a sequence of updates on the instance of $\mathcal{D}_2$. The updates are obtained by existential quantification on the changes induced by $\mathcal{J}$ and are ordered as defined by $\varphi_<$.

Finally, the query formula φ for $\mathcal{D}_1$ is the interpretation of $\mathcal{D}_2$'s query formula.

Definition 2.5. *Let $\mathcal{L}_1, \mathcal{L}_2$ be logics and τ, τ' vocabularies. Let $\mathcal{D}_1 = \mathcal{D}(\mathcal{S}_1)$ and $\mathcal{D}_2 = \mathcal{D}(\mathcal{S}_2)$ be two dynamic problems. A* dynamic $(\mathcal{L}_1, \mathcal{L}_2)$-reduction *is a triple $(\mathcal{J}_r, \mathcal{J}_u, \varphi_<)$ where $\mathcal{J}_r$ is an $\mathcal{L}_1(\tau, \tau')$-interpretation, $\mathcal{J}_u$ is an $\mathcal{L}_2(\tau, \tau')$-interpretation and $\varphi_<$ is an $\mathcal{L}_2$-formula so that $(\mathcal{J}_r, \mathcal{J}_u, \varphi_<)$ is suitable for $\mathcal{D}_1$ and $\mathcal{D}_2$. We write $\mathcal{D}_1 \leq_{(\mathcal{L}_1, \mathcal{L}_2)} \mathcal{D}_2$.*

For logics $\mathcal{L}_1$ and $\mathcal{L}_2$ that are closed under $\mathcal{L}_1$- and $\mathcal{L}_2$-interpretations, respectively, we obviously have that if $\mathcal{D}_2 \in \mathrm{Dyn}(\mathcal{L}_1, \mathcal{L}_2)$ and $\mathcal{D}_1 \leq_{\mathcal{L}_1, \mathcal{L}_2} \mathcal{D}_2$, then $\mathcal{D}_1 \in \mathrm{Dyn}(\mathcal{L}_1, \mathcal{L}_2)$.

Notice that the reduction of bipartitness to symmetric reachability given above is a dynamic reduction when we define vertices (a, u) to be smaller than (b, v) for all $u, v \in V$.

Theorem 2.6. *The following problems reduce to undirected reachability and thus are in Dyn($\mathsf{IFP}, \mathsf{FO}$).*

1. *Bipartiteness,*
2. *Clique cover-2 (CC-2): Given: An undirected graph $\mathcal{G} = (V, E)$. Problem: Can $\mathcal{G}$ be covered by two cliques?*
3. *Deterministic reachability: Given: A directed graph $\mathcal{G} = (V, E)$ and two vertices s and t. Problem: Is there a deterministic path from s to t?*
4. *k-vertex disjoint paths (k-PATHS) for fixed $k \in \mathbb{N}$: Given: An undirected graph $\mathcal{G} = (V, E)$ and two designated vertices s, t. Problem: Are there k vertex disjoint paths from s to t?*
5. *k-edge connectivity for fixed $k \in \mathbb{N}$. Given: undirected graph G and two designated vertices s and t. Problem: are there k edge disjoint paths from s to t?*

Proof. For Bipartiteness we use the interpretation from the bipartiteness example and ask whether for every $v \in V$ there is no path from v_a to v_b.

For Clique cover-2, we simply note that $\mathcal{G}$ can be covered by two cliques if and only if the complement of $\mathcal{G}$ is not bipartite.

Deterministic reachability: Let $\mathcal{G}'$ be the undirected graph that results from G when removing all edges leaving t, removing all edges leaving a vertex of degree ≥ 2 and then making the remaining edges undirected. $\mathcal{G}'$ is definable via

$$\begin{aligned}\varphi_i(x,y) =&(Exy \wedge x \neq t \wedge \forall z(Exz \rightarrow z = y))\\ &\vee (Eyx \wedge y \neq t \wedge \forall z(Eyz \rightarrow z = y)).\end{aligned}$$

When an update occurs, the following changes are induced by it. Inserting an edge (a, b) either inserts the undirected edge $\{a, b\}$ if there is no other edge (a, c), or deletes the edge $\{a, c\}$ if it is present. Deleting an edge (a, b) either deletes $\{a, b\}$ or inserts an edge $\{a, c\}$ if (a, c) is now the unique edge leaving a. Setting t to a different node may induce two changes. If a unique edge leaves the old node t, this edge is reinserted. If a unique edge leaves the new t, it is deleted. We define an order by arbitrary defining elements of an edge that is inserted as smaller than elements of an edge that is deleted.

k-vertex disjoint paths: By Menger's Theorem, for any graph $\mathcal{G} = (V, E)$ and vertices $s, t \in V$, the k-PATHS instance $(\mathcal{G}, s, t)$ has a positive answer if and only if for all $v_1, \ldots, v_{k-1} \in V \backslash \{s, t\}$, the reachability problem $(\mathcal{G}', s, t)$ has a positive answer, where $\mathcal{G}'$ is the graph obtained by deleting vertices $v_1, \ldots, v_{k-1}$ from $\mathcal{G}$. We maintain n^{k-1} copies of the graph together with all required auxiliary data to maintain undirected reachability in a $(k-1)+2$-ary relation keeping copies of the graphs with deleted vertices and two $(k-1)+3$-ary relations saving the relations F and R for each copy. Insert and delete operations induce a corresponding operation only if the affected edge is not incident with one of the deleted vertices. As all graphs are independent of each other, the reduction is not bounded first-order but bounded first-order in each component and can thus be used. The query formula asks whether for all graphs the vertex t is reachable from s.

k-edge connectivity: For any graph G and vertices s, t, the k-edge connectivity instance has a positive answer if and only if for all $k-1$ edges $e_1, \ldots, e_{k-1}$ the reachability instance G', s, t has a positive answer, where G' is the graph obtained by deleting the edges $e_1, \ldots, e_{k-1}$. Since k is constant, we universally quantify over $k-1$ edges and then check for a path in G' by composing the formula for undirected reachability (for a single deletion) $k-1$ times. □

There has been much research on reduction concepts respecting dynamic classes. The main difference to the concept appropriate for our setting results from the fact that reductions in the literature have to deal with ordered classes only, whereas our reductions have to work in the more general, purely logical setting. Patnaik and Immerman [20] consider bounded expansion first-order reductions, that is first-order interpretations such that each tuple in a relation of the input structure affects at most a constant number of tuples in the output structure. This easily generalizes to bounded expansion $\mathcal{L}$-reductions for any logic $\mathcal{L}$. They further introduce bounded expansion reductions with pre-computation, which allow a polynomial pre-computation. Our setting is very similar to that of [21]. The approach in [21] of dealing with the initial structure with a strong logic first and handling the updates with a simpler logic directly carries over to the reduction concept. Both settings are restricted to ordered structures and thus evade the problem of providing an order on the updates (there is always the order provided by the structure). Note that in [15] and [21] the reduction does not depend on the structure it is working on. Each operation on the instance of $\mathcal{D}_1$ must induce a unique set of operations on the instance of $\mathcal{D}_2$. Thus the reduction of deterministic reachability to symmetric reachability in Theorem 2.6 is not a valid reduction in their setting. In [15] Hesse and Immerman present complete problems for dynamic ordered classes in a setting which is different but still closely related to ours.

Dong et. al do not explicitly define a reduction concept. In [6] they state that generally IES are not scalable, i.e. in general, IES cannot deal with a fixed bounded number of updates at the same time. The reason for this is that there is possibly no definable order on the updates. To overcome this, they introduce deterministic IES, where the order of updates is not relevant. We discuss this concept in the next section.

3. HISTORY INDEPENDENT DYNAMIC SYSTEMS

An Incremental Evaluation System is called history independent (deterministic in [6] and memoryless in [20]) if the auxiliary relations maintained in the dynamic process are independent of the order of the updates but depend only on the structure that results from the updates. The auxiliary relations are at all times described by the formulae that are used to initialize the dynamic process. The refined notion obtained by separating initial logic and update logic gives a good insight on the complexity of the auxiliary relations. The main motivation to study history independent IES is that they allow for a simpler reduction concept and can deal with any fixed number of updates happening at the same time. No order has to be definable on the induced changes of an operation as any order of applying changes yields the same auxiliary relations.

It was posed as an open question by [6] whether there is a history independent incremental system with FO-updates for reachability in undirected graphs or bipartiteness. We give a positive answer.

Definition 3.1. *Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be logics and let I be an $(\mathcal{L}_1, \mathcal{L}_2)$-IES for a dynamic problem. Let $\mathcal{J}$ be the initial interpretation of I. The IES is called* history independent, *if for all update sequences $w = v\delta$ of length $i > 0$, where δ is an update with corresponding update interpretation $\mathcal{J}_\delta$ and $\mathfrak{B}_i$ is the result of the dynamic process of applying v to $\mathfrak{A}$ we have*

$$\mathfrak{B}_{i+1} = \mathcal{J}(w(\mathfrak{A})) = \mathcal{J}_\delta(w(\mathfrak{A}) \sqcup (\mathfrak{B}_i \restriction \tau')).$$

Theorem 3.2. *Reachability on symmetric graphs is in history independent Dyn*(IFP, FO).

Proof. Let $\mathcal{G} = (V, E)$ be a graph. The IES maintains shortest paths and 'numbers' up to the diameter of $\mathcal{G}$. These numbers are realized by tuples of elements, where a tuple (x, y) stands for the length of a shortest path between x and y and it represents ∞ if no path exists. Write $|xy|$ for (x, y) if it appears in this context. We also maintain equality, addition and order on the numbers. We use $=$ and $<$ in infix notation and use $+$ as a function even though it is internally treated as a relation. The predicate $Fuvxy$ means (u, v) lies

on a shortest path from x to y and we have $Fvvvv$ for all vertices v.

Initialization. We present the initialization on an empty graph via a first-order interpretation. Obviously, the above described relations are IFP-definable when starting with an arbitrary graph. When starting the dynamic process with an empty graph, there are only the numbers 0 represented by tuples (a, a) for all elements a, and ∞ represented by (a, b) for elements $a \neq b$. The numerical predicates are initialized such that the following holds: $0 = 0$, $\infty = \infty$, $0 < \infty$, $0 + 0 = 0$, $0 + \infty = \infty$, $\infty + 0 = \infty$ and $\infty + \infty = \infty$. We thus have

$$|x_1y_1| = |x_2y_2| \Leftrightarrow (x_1 = y_1 \wedge x_2 = y_2) \vee (x_1 \neq y_1 \wedge x_2 \neq y_2)$$

$$|x_1y_1| < |x_2y_2| \Leftrightarrow x_1 = y_1 \wedge x_2 \neq y_2.$$

$$|x_1y_1| + |x_2y_2| = |x_3y_3| \Leftrightarrow (x_1 = y_1 \wedge x_2 = y_2 \wedge x_3 = y_3) \vee ((x_1 \neq y_1 \vee x_2 \neq y_2) \wedge x_3 \neq y_3).$$

$$Fuvxy \equiv u = v = x = y.$$

Insert$_E$. Assume that the undirected edge $\{a, b\}$ is inserted into the graph. We describe all necessary updates that have to be performed by formulae φ_1, φ_2 and φ_3 which we then compose to the update formula for the predicate F. For all times and all vertices v we want that $Fvvvv$ holds. This is guaranteed by $\varphi_1 = (u = v = x = y)$.

Inductively, we may assume that $|xy|$ is the length of a shortest path from x to y. If it holds that $|xa| + 1 + |by| \geq |xy|$, then one does not obtain a shorter path by walking the new edge $\{a, b\}$ in direction (a, b). Analogously, if $|xb| + 1 + |ay| \geq |xy|$ it is no shortcut to walk the edge in direction (b, a). If using neither direction introduces a shortcut, we keep the path by formula φ_2. Note that evaluating $|xa| + 1 + |by| \geq |xy|$ requires that at least one edge (c, d) is already available to represent the number 1. The newly added edge (a, b) may represent any number or ∞ at this point.

$$\varphi_2 = (\exists c \exists d(E^{old}cd \wedge (Fuvxy \wedge |xa| + |cd| + |by| \geq |xy| \wedge |xb| + |cd| + |ay| \geq |xy|))).$$

If on the other hand $|xa| + |by| < |xy|$, then $|xa| + 1 + |by| \leq |xy|$ and there is a new path of minimal length, possibly of the same length as an existing path of minimal length. Old paths of the same length are kept by φ_2. Observe that $|xa| + |by| < |xy|$ implies that there are paths from x to a and from b to y, since $\infty \not< \infty$. For the new path, we concatenate shortest paths from x to a, the edge (a, b) and shortest paths from b to y. Clearly, paths created that way are shortest paths, especially no circles are created and only one of (a, b) or (b, a) is used. The new paths are created by φ_3.

$$\varphi_3 = (Fuvxa \vee (u = a \wedge v = b) \vee Fuvby) \wedge |xa| + |by| < |xy|.$$

Finally we obtain the new predicate F via

$$F'uvxy \equiv \varphi_1 \vee \exists a \exists b(Eab \wedge \neg E^{old}ab \wedge (\varphi_2 \vee \varphi_3))$$

Note that quantification in the form $\exists a \exists b(Eab \wedge \neg E^{old}ab \wedge \varphi)$ results in interpreting $\{a, b\}$ once as (a, b) and once as (b, a). This can be seen as calculating the new shortest in parallel and then choosing the right direction.

A comment on evaluating sums of numbers is in place. Numbers are only available up to the length of a longest shortest path. Thus a sum $a_1 + \ldots + a_k$ may not exist as a number and cannot be evaluated. But all such terms appear only in equations or inequalities of the form $a_1 + \ldots + a_k = b_1 + \ldots + b_s$ or $a_1 + \ldots + a_k < b_1 + \ldots + b_s$ and can be rewritten, e.g.

$$\begin{aligned} &a + b > c + d \Leftrightarrow \\ &\quad (a < c \wedge b > d \wedge \exists e \exists f(a + e = c \wedge d + f = b \wedge f > e) \vee \\ &\quad (a < d \wedge b > c \wedge \ldots \end{aligned}$$

To update the arithmetic predicates we have to check whether a tuple $|xy|$ changes its meaning. From the update formula for shortest paths above we know that a tuple $|xy|$ either keeps its value or changes its value to $|xa| + 1 + |by|$. We begin with updating the equality predicate. Again we handle the occuring situations by several subformulae which will be composed to obtain the new predicate.

We can express $0 = 0$ without accessing any predicates by the formula $\eta_1 := (x_1 = y_1 \wedge x_2 = y_2)$. We use the newly updated predicate F' to express $\infty = \infty$ in formula η_2:

$$\eta_2 = \forall u_1 \forall v_1 \forall u_2 \forall v_2(\neg(u_1 = v_1 = x_1 = y_1) \rightarrow \neg F'u_1v_1x_1y_1 \wedge \neg(u_2 = v_2 = x_2 = y_2) \rightarrow \neg F'u_2v_2x_2y_2).$$

We use formulae η_3 to η_6 to replace numbers by their new meaning if necessary. η_3 handles the case that $|x_1y_1|$ and $|x_2y_2|$ keep their value. η_4 deals with the case that only $|x_1y_1|$ changes its value while $|x_2y_2|$ keeps its value. η_5 is analogous for the case that only $|x_2y_2|$ changes its value. η_6 deals with the case that both distances change. Note that if there is no edge present in the graph, all equalities are described by η_1 and η_2.

$$\begin{aligned} \eta_3 = &\bigwedge_{i \in \{1,2\}} |x_iy_i| \leq |x_ia| + |cd| + |by_i| \wedge \\ &|x_iy_i| \leq |x_ib| + |cd| + |ay_i| \wedge |x_1y_1| = |x_2y_2|. \end{aligned}$$

$$\begin{aligned} \eta_4 = &|x_1y_1| > |x_1a| + |cd| + |by_1| \wedge \\ &|x_2y_2| \leq |x_2a| + |cd| + |by_2| \wedge \\ &|x_2y_2| = |x_1a| + |cd| + |by_1|. \end{aligned}$$

$$\begin{aligned} \eta_5 = &|x_2y_2| > |x_2a| + |cd| + |by_2| \\ &\wedge |x_1y_1| \leq |x_1a| + |cd| + |by_1| \wedge \\ &|x_1y_1| = |x_2a| + |cd| + |by_2|. \end{aligned}$$

$$\begin{aligned} \eta_6 = &|x_1y_1| > |x_1a| + |cd| + |by_1| \\ &\wedge |x_2y_2| > |x_2a| + |cd| + |by_2| \wedge \\ &|x_1a| + |by_1| = |x_2a| + |by_2|. \end{aligned}$$

Finally we obtain the update formula for equality as

$$\begin{aligned} |x_1y_1| =' |x_2y_2| \Leftrightarrow\ &\eta_1 \vee \eta_2\ \vee \exists a \exists b(Eab \wedge \neg E^{old}ab \\ &\wedge [\exists c \exists d(E^{old}cd \wedge (\eta_3 \vee \eta_4 \vee \eta_5 \vee \eta_6))]) \end{aligned}$$

Order is changed similarly to equality. We have a formula ψ_1 which says $0 < i$ for all $i \neq 0$ and a formula ψ_2 which

says $i < \infty$ for all $i \in \mathbb{N}$.

$$\psi_1 = (x_1 = y_1 \wedge x_2 \neq y_2).$$

$$\psi_2 = \exists u_1 \exists v_1 \forall u_2 \forall v_2 (F' u_1 v_1 x_1 y_1 \wedge \neg F' u_2 v_2 x_2 y_2).$$

Formula ψ_3 deals with the case that $\{a, b\}$ is inserted to the empty graph, that is it says $|ab| = 1 < \infty$.

$$\psi_3 = \neg\exists c \exists d (E^{old} cd \wedge (x_1 = a \wedge y_1 = b \wedge \neg\exists u \exists v F' uv x_2 y_2)).$$

We do not explicitely spell out ψ_4 as it is very similar to η_3 to η_6 above. We combine the formulae to obtain the updated order predicate.

$$|x_1 y_1| <' |x_2 y_2| \Leftrightarrow \psi_1 \vee \psi_2 \vee \exists a \exists b (Eab \wedge \neg E^{old} ab \wedge (\psi_3 \vee \psi_4)).$$

Finally we give the update formula for addition. We express $0 + 0 = 0$ with formula ϑ_1, ϑ_2 says $0 + \infty = \infty$, $\infty + 0 = \infty$ and $\infty + \infty = \infty$.

$$\vartheta_1 = (x_1 = y_1 \wedge x_2 = y_2 \wedge x_3 = y_3).$$

$$\begin{aligned}\vartheta_2 = \forall u_1 \forall v_1 \forall u_3 \forall v_3 (\neg(u_1 = v_1 = x_1 = y_1) \rightarrow \neg F' u_1 v_1 x_1 y_1 \wedge \\ \neg(u_3 = v_3 = x_3 = y_3) \rightarrow \neg F' u_3 v_3 x_3 y_3) \vee \\ \forall u_2 \forall v_2 \forall u_3 \forall v_3 (\neg(u_2 = v_2 = x_2 = y_2) \rightarrow \neg F' u_2 v_2 x_2 y_2 \wedge \\ \neg(u_3 = v_3 = x_3 = y_3) \rightarrow \neg F' u_3 v_3 x_3 y_3).\end{aligned}$$

Formula ϑ_3 deals with the case that no edge was present when $\{a, b\}$ was inserted, we have $0 + |ab| = |ab|$ and $|ab| + 0 = |ab|$.

$$\begin{aligned}\vartheta_3 = \neg\exists c \exists d (E^{old} cd \wedge \\ (x_1 = y_1 \wedge x_2 = a \wedge y_2 = b \wedge x_3 = a \wedge y_3 = b) \vee \\ (x_2 = y_2 \wedge x_1 = a \wedge y_1 = b \wedge x_3 = a \wedge y_3 = b)).\end{aligned}$$

When edges are present, addition is defined by ϑ_4 similar as above by replacing numbers by their new meaning if necessary. The update formula for addition is

$$\begin{aligned}|x_1 y_1| +' |x_2 y_2| = |x_3 y_3| \Leftrightarrow \vartheta_1 \vee \vartheta_2 \vee \\ \exists a \exists b (Eab \wedge \neg E^{old} ab \wedge (\vartheta_3 \vee \vartheta_4)).\end{aligned}$$

Delete$_E$. Key to handling deletions of edges is the availability of the arithmetic predicates. With their help we can determine new shortest paths after the deletion of an edge. Shortest paths from a vertex x induce a corresponding acyclic graph G_x defined by $(y, z) \in G_x \Leftrightarrow \exists u Fyzxu \wedge y \neq z$, that is (y, z) is an edge of G_x if it lies on a shortest path from x to some vertex u. We will need the transitive closure R_x of G_x for each x as an intermediate relation. R_x is implicit in F since we have $Rxyz \equiv \exists u (Fuyxy \wedge Fuyxz)$ which holds true if z is reachable from y in G_x. As G_x is acyclic for each x we can express R_x after the deletion of any edge again by the result of [8].

Maintenance of the auxiliary relations is based on the following combinatorial arguments. For each pair u, v of vertices, if there is a path from u to v in G_x for some x, then this path is a shortest path from u to v. Note that there may not exist a path from u to v in G_x for every x even though u and v are connected in $\mathcal{G}$. When deleting $\{a, b\}$, we first delete (a, b) or (b, a) from G_x for every x using the result of [8]. Assume that (a, b) is deleted. Denote the graph obtained by deleting (a, b) by G'_x. Denote the vertices which are reachable from x in G'_x by R'_x and those which are no longer reachable by N'_x. All paths remaining in G'_x are shortest paths between the corresponding vertices. Again we have to find alternative paths if a vertex is still reachable in $\mathcal{G}$ but not in G'_x after the deletion of (a, b). The interesting case is again when b is no longer reachable in G_x after deletion of (a, b). Every vertex which was reachable before the deletion of (a, b) is also reachable after the deletion if the following holds. From b a vertex u is reachable in G_x before the deletion of (a, b), which is either (1) reachable after the deletion of (a, b) or (2) connected to a vertex v reachable after the deletion of (a, b) via edge (u, v) which is not in G_x. The above conditions then spell (1) from b a vertex of R'_x is reachable or (2) from b a vertex w in N'_x is reachable which is connected to a vertex from R'_x via an edge in E.

Assume that after the deletion of (a, b) no path from x to b exists in G'_x, thus new paths have to be defined. Let $y \in N'_x$ and let π be a shortest path from x to y in $\mathcal{G}$. On π there is a first visit of a vertex w in N'_x. We show that from w the path runs straight to y without leaving N'_x and in particular without using edge (a, b) or (b, a). Assume otherwise, that is, after w there is another visit of a vertex z in R'_x. Then π from x to z is as short as some path from x to z in G'_x and thus π is minimal from x to z without using (a, b). As w is not reachable after the deletion of (a, b), there is a shorter path from x to w via (a, b) than π from x to w. Then this path from x via (a, b) to w and π from w to z is a shorter path from x to z than π is from x to z, a contradiction to $z \in R'_x$.

Thus a shortest path from x to y is the concatenation of a minimal path from x to a vertex $v \in R'_x$ which is connected to a vertex $w \in N'_x$ and a minimal path from this w to y. To find a new path we want to use the path of minimal length of all paths of this form. Analogous to the argumentation above, we show that all paths of this form are valid, that is, no such path uses (a, b) or (b, a) and thus proposes a shortest path which no longer exists.

We obtain the following update formula for F. Write $Rxyz$ as an abbreviation for the predicate obtained by applying the formula for reachability in acyclic graphs from [8] to $\exists u (Fuyxy \wedge Fuyxz)$.

$$\begin{aligned}F' uvxy \equiv (Fuvxy \wedge Rxxy \vee \\ (\neg Rxxy \wedge \exists r \exists s (Rxxr \wedge \neg Rxxs \wedge Ers \wedge \\ (Fuvxr \vee (u = r \wedge v = s) \vee Fuvsy) \wedge \\ \forall r' \forall s' (Rxxr' \wedge \neg Rxxs' \wedge Er's' \rightarrow \\ |xr| + |sy| \leq |xr'| + |s'y|)))).\end{aligned}$$

For the arithmetic predicates we observe that we can proceed exactly as for the insert operations: Determine in FO whether the length of a path changes and define its new length as in the update formula for F above as $\min\{|xv| + 1 + |wb| : v \in R'_x, w \in N'_x, Evw\}$ and 0 if no such w exists. The query formula is $\varphi = \exists x \exists y Fxyst$. □

We adapt the idea of using pairs of vertices as numbers to modify the algorithm of Etessami [12] and obtain a history independent IES with first-order plus counting updates for dynamic tree isomorphism. Tree isomorphism is defined for structures of vocabulary (E_1, E_2, c_1, c_2), where E_1 and E_2 are binary and c_1, c_2 are constants. (A, E_1, E_2, c_1, c_2) is a positive instance of the tree isomorphism problem if the tree

induced by E_1 rooted at c_1 and the tree induced by E_2 rooted at c_2, respectively, are isomorphic.

THEOREM 3.3. *Tree isomorphism is contained in the class Dyn*(IFP+C, FO+C).

PROOF. The dynamic algorithm of Etessami [12] uses the following auxiliary relations which can be initialized with IFP+C: $path(x, y)$ denoting the reflexive transitive closure of the edge relation, $dist(x, y, d)$ denoting distances in the forest, restricted to finite distances. This relation is modified such that it uses numbers and arithmetic as in our incremental evaluation system for symmetric reachability. $onpath(x, y, z)$ means $path(x, y)$ and z is on the unique shortest path from x to y and T-ISO(x, x', y, y') means that the subtrees rooted at x' and y' in the trees rooted at x and y, respectively, are isomorphic. All auxiliary relations satisfy the above specifications throughout the dynamic process and are thus history independent.

When the dynamic process starts with an empty structure or is applied to ordered structures, the need for counting can be completely eliminated. Etessami uses a further relation *sibling-iso-count* to dynamically maintain the count of isomorphic siblings. This cannot be done in the unordered setting, as there may not be enough types of element tuples to represent the required numbers. We will show in the next section that the above bounds for tree isomorphism are optimal. □

Recall definition 2.5 of a dynamic reduction. We called a triple $(\mathcal{J}_r, \mathcal{J}, \varphi_<)$ suitable for dynamic problems $\mathcal{D}_1$ and $\mathcal{D}_2$ if for all $\mathfrak{A}$

$$(\mathfrak{A}, w) \in \mathcal{D}_1 \Leftrightarrow (\mathcal{J}_r(\mathfrak{A}), h^<_{\mathcal{J}}(w)) \in \mathcal{D}_2,$$

where $h^<_{\mathcal{J}}(w)$ denotes the sequence of updates that is obtained by bringing the induced updates in the lexicographical order $<$ induced by $\varphi_<$. As history independent IES do not depend on the update order we do no longer require that an order has to be definable on the updates. A tuple $(\mathcal{J}_r, \mathcal{J})$ is suitable for $\mathcal{D}_1$ and $\mathcal{D}_2$ if for all $\mathfrak{A}$, all update sequences w and any linear order $<$ on A

$$(\mathfrak{A}, w) \in \mathcal{D}_1 \Leftrightarrow (\mathcal{J}_r(\mathfrak{A}), h^<_{\mathcal{J}}(w)) \in \mathcal{D}_2.$$

DEFINITION 3.4. *Let $\mathcal{L}_1, \mathcal{L}_2$ be logics and τ, τ' vocabularies. Let $\mathcal{D}_1 = \mathcal{D}(\mathcal{S}_1)$ and $\mathcal{D}_2 = \mathcal{D}(\mathcal{S}_2)$ be dynamic problems. A history independent $(\mathcal{L}_1, \mathcal{L}_2)$-reduction from $\mathcal{D}_1$ to $\mathcal{D}_2$ is a tuple $(\mathcal{J}_r, \mathcal{J}_u)$ of an $\mathcal{L}_1(\tau, \tau')$-interpretation $\mathcal{J}_r$ and an $\mathcal{L}_2(\tau, \tau')$-interpretation $\mathcal{J}_u$ with the bounded expansion property such that $(\mathcal{J}_r, \mathcal{J}_u)$ is suitable for $\mathcal{D}_1$ and $\mathcal{D}_2$. We write $\mathcal{D}_1 \leq^{det}_{(\mathcal{L}_1, \mathcal{L}_2)} \mathcal{D}_2$.*

THEOREM 3.5. *Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be closed under $\mathcal{L}_1$- and $\mathcal{L}_2$-interpretations, respectively. Let $\mathcal{D}_2$ be in history independent Dyn$(\mathcal{L}_1, \mathcal{L}_2)$ and $\mathcal{D}_1 \leq^{det}_{\mathcal{L}_1, \mathcal{L}_2} \mathcal{D}_2$. Then $\mathcal{D}_1$ is in history independent Dyn$(\mathcal{L}_1, \mathcal{L}_2)$.*

COROLLARY 3.6. *All properties from Theorem 2.6 are in history independent Dyn*(IFP, FO).

4. NON-DEFINABILITY RESULTS FOR DYNAMIC PROBLEMS

In the following section we first show that the above auxiliary relations used to maintain the transitive closure of an undirected graph with first-order updates do not suffice to maintain the transitive closure of a directed graph with first-order updates. To show this we use a technique which was already used in [3, 5] to show that the transitive closure cannot be maintained with help of only the transitive closure itself and additional unary relations. The idea is to use the update-formulae of an assumably existing IES on a graph where all auxiliary relations are already first-order definable, modify this graph by performing definable updates and obtain as a result of the combined formulae a first-order formula defining the transitive closure of a graph on which the transitive closure is known to be undefinable.

We then show that the equal cardinality query is not in Dyn$(\mathcal{L}, \mathsf{FO})$ for any logic $\mathcal{L}$, whatever auxiliary relations are used. The reason for this is the inability of logics to define sufficiently many numbers in the initialization step on highly symmetric structures. Thus no IES can keep track of the difference between sufficiently large structures. It follows that tree isomorphism is not in Dyn$(\mathcal{L}, \mathsf{FO})$ for any logic $\mathcal{L}$, as equal cardinality reduces to it via (FO, FO)-reductions.

THEOREM 4.1. *There is no incremental evaluation system with first-order updates for the transitive closure of a graph using the same auxiliary relations as for the symmetric transitive closure in Theorem 3.2 together with an arbitrary number of unary auxiliary relations.*

PROOF. Assume towards a contradiction that there is an IES for the transitive closure using the same formulae as in Theorem 3.2 to initialize its auxiliary relations, together with k formulae to initialize k additional unary auxiliary relations. Consider a structure consisting of two large disjoint cycles together with two additional elements which are connected in one direction via an edge, i.e. the structure $\mathcal{G} = (V, E)$ where $V = \{1, \ldots, 2n, 2n+1, 2n+2\}$ and $E = \{(i, i+1) : 1 \leq i < n, n+1 \leq i < 2n\} \cup \{(n, 1), (2n, n+1)\} \cup \{(2n+1, 2n+2)\}$ for sufficiently large n. It follows from a simple locality argument that reachability between vertices on the cycles is not first-order definable. We define via first-order logic the graph $\mathcal{G}'$ with additional edges $\{(i, 2n+1) : 1 \leq i < 2n\} \cup \{(2n+2, i) : 1 \leq i < 2n\}$. On $\mathcal{G}'$ all auxiliary relations from Theorem 3.2 are first-order definable, as the maximal distance of two vertices is three. Also there are only finitely many ways to define a coloring of the graph by logical means, as there are only 3 isomorphism types of single elements. Thus, there is an FO-interpretation which initializes all auxiliary relations on $\mathcal{G}'$ as the IFP-initialization for the IES would have done. We now use the update-interpretation for deletes to delete the edge $(2n+1, 2n+2)$, which is definable in FO, from $\mathcal{G}'$. From this, one can easily define the transitive closure of the original graph $\mathcal{G}$. Concatinating those first-order formulae yields a first-order formula which defines the transitive closure of the vertices on the cycles, a contradiction. □

THEOREM 4.2. *Equal cardinality of unary relations is not in Dyn$(\mathcal{L}, \mathsf{FO})$ for any logic $\mathcal{L}$.*

PROOF. We consider structures of vocabulary $\tau = (U_1, U_2)$ and show that we cannot dynamically maintain whether $|U_1| = |U_2|$. Assume there is an IES which uses k auxiliary relations. Via a dynamic reduction with two parameters we can pass to an IES with a single auxiliary relation of larger arity. Without loss of generality we may thus assume that we have an IES $(\mathcal{J}_i, \mathcal{J}^{ins}_R, \mathcal{J}^{del}_R, \varphi_q)$ based on the single auxiliary relation R of arity r. The interpretations $\mathcal{J}^{ins}_R$ and

$\mathcal{J}_R^{del}$ consist of a single formula φ_{ins} and φ_{del}, respectively. We want to show that there are two initial structures on which the dynamic process proceeds in a very similar way if the same update sequence is applied to both structures. The resulting structures are in fact m_q-equivalent, where m_q is the quantifier rank of φ_q and thus φ_q agrees on both structures. But the structures are chosen in a way that they disagree on the number of elements that need to be inserted to U_2 to satisfy $|U_1| = |U_2|$. We thus obtain a contradiction, proving that no IES for equal cardinality exists.

Let m_i be the quantifier rank of φ_{ins} and let $m = r + 1 + \max\{m_i, m_q\}$. For a sequence w of updates and a τ-structure $\mathfrak{A}$ we write $\mathfrak{A}_w$ for $w(\mathfrak{A})$.

Our first observation is that the atomic type of an element determines its behavior as long as it was not part of an update operation. Call the set of elements that were part of an update operation the accessed domain of a structure.

Claim 4.3. *Let τ be a vocabulary that contains unary predicates only. Let w be a sequence of operations and let $(a_1, \dots, a_r) \in A^r$ such that $\overline{a} = (a_{i_1}, \dots, a_{i_t})$ are not in the accessed domain. Let $\overline{b} = (b_{i_1}, \dots, b_{i_t})$ not in the accessed domain be of the same atomic type as $\overline{a}$ (w.r.t. the input relations). Then $(a_1, \dots, a_r) \in R^{\mathfrak{A}_w}$ if and only if $(a_1, \dots, a_r)[a_{i_1}/b_{i_1}, \dots, a_{i_t}/b_{i_t}] \in R^{\mathfrak{A}_w}$.*

Proof. It is easily shown via induction on the length of w that the mapping which swaps $a_{i_j} \in \overline{a}$ with $b_{i_j} \in \overline{b}$ is an automorphism of $(\mathfrak{A}_w, R^{\mathfrak{A}_w})$. □

Thus, the initial auxiliary relation of any structure of unary signature is characterized by the set of atomic types of the tuples it contains. Call $\mathrm{atp}_{\mathfrak{A}}(R^{\mathfrak{A}}) := \{\mathrm{atp}(\overline{a}) : \mathfrak{A} \models R\overline{a}\}$ the atomic type of a relation $R^{\mathfrak{A}}$. As there are only finitely many atomic types of tuples, the number of relation types is finite, too.

We conclude that there is $M \in \mathbb{N}$ such that there are two structures $\mathfrak{A}, \mathfrak{B}$ with $A = B = \{1, \dots, 3M\}$, $U_1^{\mathfrak{A}} = \{1, \dots, n_A\}$, $U_1^{\mathfrak{B}} = \{1, \dots, n_B\}$ for some n_A, n_B with $m < n_A < n_B \leq M$, $U_2^{\mathfrak{A}} = U_2^{\mathfrak{B}} = \emptyset$, such that the initial auxiliary relations $R^{\mathfrak{A}}$ and $R^{\mathfrak{B}}$ have the same type. All sizes are chosen such that update and query formula cannot distinguish between the two structures. Intuitively, a formula of quantifier rank q cannot distinguish between the number of witnesses of some element type, if there are more than q witnesses.

Claim 4.4. *Let $\mathfrak{A}$, $\mathfrak{B}$ and m be as described above. Let w be the sequence of insert operations inserting the elements $2M + 1, \dots, 2M + n_A$ into U_2 in their natural order. Then $(\mathfrak{A}_w, R^{\mathfrak{A}_w})$ and $(\mathfrak{B}_w, R^{\mathfrak{B}_w})$ are m-equivalent.*

Proof. Denote by w_i the prefix of length i of w. We show via induction on i that the identity mapping on $U = \{1, \dots, n_A, M, M+1, \dots, 3M\}$ is a partial isomorphism from $(\mathfrak{A}_{w_i}, R^{\mathfrak{A}_{w_i}})$ to $(\mathfrak{B}_{w_i}, R^{\mathfrak{B}_{w_i}})$.

The statement holds for $i = 0$ as both $\mathfrak{A}$ and $\mathfrak{B}$ have more than m elements of each atomic 1-type, $\mathfrak{A}$ and $\mathfrak{B}$ have the the same initial relation type and by Claim 4.3.

Now assume the statement for some $1 < i < n$ and element $2M + i + 1$ is inserted into U_2. We have to show that $R^{\mathfrak{A}_{w_{i+1}}}\overline{a} \Leftrightarrow R^{\mathfrak{B}_{w_{i+1}}}\overline{a}$ for all $\overline{a} \in U^r$. Recall that $R^{\mathfrak{A}_{w_{i+1}}}$ and $R^{\mathfrak{B}_{w_{i+1}}}$ are defined by φ_{ins} of quantifier-rank q_i. We introduce a new constant c, which is interpreted as $2M+i+1$ in both structures and show that $(\mathfrak{A}_v, R^{\mathfrak{A}_v}, c^{\mathfrak{A}}, \overline{a})$ and $(\mathfrak{B}_v, R^{\mathfrak{B}_v}, c^{\mathfrak{B}}, \overline{a})$ are q_i-equivalent for all $\overline{a} \in U^r$. This proves the statement. In the q_i-round Ehrenfeucht-Fraïssé game, the duplicator can exactly copy all moves on U. On all other parts she can place her pebbles inside U according to atomic type with respect to the input relations. This is possible by the choice of the structure sizes ($m \geq q_i + r + 1$). To see that this is a winning strategy, we may move pebbles outside U to elements of the same type with respect to the input relations by Claim 4.3 without changing the atomic type of the pebbled elements. We move them such that we obtain the identity mapping on U, which is a partial isomorphism by assumption.

The proof of the claim is analogous to the proof in the induction step of the above statement. □

We have thus shown that the dynamic process runs very similarly on structures of different sizes. To finally show that the equal cardinality query is not maintainable we choose two structures $\mathfrak{A}, \mathfrak{B}$ as above. Applying the operation w to both structures yields two m_q-equivalent structures. Thus the dynamic algorithm produces the same answer on both structures. But as the sizes of $U_1^{\mathfrak{A}}$ and $U_1^{\mathfrak{B}}$ differ, $\mathfrak{A}_w$ satisfies the equal cardinality query and $\mathfrak{B}_w$ does not.

Corollary 4.5. $\mathsf{FO{+}C} \not\subseteq \mathit{Dyn}(\mathcal{L}, \mathsf{FO})$ *for any logic $\mathcal{L}$.*

Corollary 4.6. *Tree isomorphism is not in $\mathit{Dyn}(\mathcal{L}, \mathsf{FO})$ for any logic $\mathcal{L}$.*

Proof. Equal cardinality reduces to tree isomorphism via a dynamic $(\mathsf{FO}, \mathsf{FO})$-reduction with parameters a_1, a_2. Create two trees of depth 1, where each node v with $U_i v$ is attached to root a_i. □

Our notion of a history independent $(\mathcal{L}, \mathsf{FO})$-IES is closely related to the notion of deterministic *FOIES* from [6]. If there is a history independent $(\mathcal{L}, \mathsf{FO})$-IES for a query then there is a deterministic *FOIES* for the query. If on the other hand there is a deterministic *FOIES* for a query there is a history independent $(L_{\infty,\omega}, \mathsf{FO})$-IES for the query. Thus if there is no $(\mathcal{L}, \mathsf{FO})$-IES for a query for any logic $\mathcal{L}$, there is no deterministic *FOIES* either.

Corollary 4.7. *For equal cardinality and for tree isomorphism there are no deterministic FOIES.*

It was shown by Etessami [12] that an order and arithmetic can be built on all elements that were part of an update operation. Especially when the dynamic process starts with the empty structure, order and arithmetic are available on all relevant parts of the structure. In this case tree isomorphism is dynamically maintainable. It was shown in [13] that $\mathsf{FO{+}TC{+}C}$ does not suffice to express tree isomorphism and Corollary 4.6 shows that FO updates do not suffice to maintain sufficient information in the dynamic setting. Thus the bounds from Theorem 3.3 on unordered structures are in a sense optimal.

It is open whether the need for counting quantifiers can be eliminated completely when starting with the empty structure or when dealing with ordered structures. In other words it is open whether $\mathsf{ACC}^0 \subseteq \mathrm{Dyn}(\mathcal{C}, \mathsf{AC}^0)$ or $\mathsf{TC}^0 \subseteq \mathrm{Dyn}(\mathcal{C}, \mathsf{AC}^0)$ for any complexity class $\mathcal{C}$.

5. CONCLUSION AND FUTURE WORK

We adapted the dynamic setting of [21] to work on unordered structures and investigated whether the known results transfer to this setting. We showed that many of the known incremental evaluation systems for dynamic problems can be translated to work in the unordered setting. These include reachability in symmetric graphs, bipartiteness, k-vertex disjoint path and more. We even provided history independent incremental evaluation systems for all of the above problems. We showed how to modify the algorithm of [12] to handle tree isomorphism in a history independent fashion with FO+C-updates after an IFP+C-initialization. Furthermore we were able to show several lower bounds based on the restrictions imposed by purely logical initialization. These include the impossibility of handling the equal cardinality query and the tree isomorphism query with FO-updates.

The query that, for its practical relevance, receives most attention in the literature is the transitive closure of a directed graph. It is widely believed that it is not possible to maintain this property with FO-updates, yet a proof of this conjecture remains a major open problem. We believe that, especially in the context of history independent IES, a first step towards solving the problem is to better understand the (logical) complexity of required auxiliary relations. As transitive closure is expressible in Datalog, it is an interesting question whether auxiliary relations initialized with Datalog suffice to maintain it with FO-updates. We conjecture that they do not and preservation properties for Datalog may help to establish this result. Other candidate logics that do not allow the construction of the auxiliary relations we used to maintain the symmetric transitive closure are Datalog($\neq$) and stratified linear Datalog (which is equivalent to FO+TC). The result of Hesse [14] showed that the transitive closure is maintainable with TC^0-updates on ordered structures but this result does not translate to the unordered setting. It is open whether FO+C or even FO+STC-updates suffice to maintain the transitive closure in the unordered setting.

Lots of focus has been on the canonical updates which result in small changes to the considered structures. We think that it is interesting to consider updates that are induced by first-order formulae. On the one hand one can consider formulae which induce updates directly to the structure, i.e. consider updates that change all tuples with the property defined by the formula. On the other hand one can perform canonical updates to one structure and consider the changes that are induced on a first-order interpreted structure. For example the transitive closure over a general formula can be read as the transitive closure over the edge relation of an interpreted graph. When considering the transitive closure over a formula which does not have the bounded-expansion property, as required for dynamic reductions, a single update to the original graph induces many changes on the interpreted graph. An IES cannot quantify all the changes at once, a fact which can possibly be used to prove that the transitive closure over arbitrary formulae is not maintainable.

6. REFERENCES

[1] David A. Mix Barrington, Neil Immerman, and Howard Straubing. On uniformity within NC. *Journal of Computer and System Sciences*, 41(3):274–306, 1990.

[2] Guozhu Dong, Leonid Libkin, and Limsoon Wong. Incremental recomputation in local languages. *Information and Computation*, 181:2003, 2001.

[3] Guozhu Dong and Jianwen Su. Increment boundedness and nonrecursive incremental evaluation of datalog queries. In *In LNCS 893: Proceedings of 5th International Conference on Database Theory*, pages 397–410. Springer-Verlag, 1995.

[4] Guozhu Dong and Jianwen Su. Incremental and decremental evaluation of transitive closure by first-order queries. *Inf. Comput.*, 120(1):101–106, 1995.

[5] Guozhu Dong and Jianwen Su. Space-bounded FOIES (extended abstract). In *PODS '95: Proceedings of the fourteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 139–150, New York, NY, USA, 1995. ACM.

[6] Guozhu Dong and Jianwen Su. Deterministic FOIES are strictly weaker. *Annals of Mathematics and Artificial Intelligence*, 19(1-2):127–146, 1997.

[7] Guozhu Dong and Jianwen Su. Arity bounds in first-order incremental evaluation and definition of polynomial time database queries. In *Journal of Computer and System Sciences*, volume 57, pages 289–308, 1998.

[8] Guozhu Dong, Jianwen Su, and Rodney Topor. First-order incremental evaluation of datalog queries. In *Annals of Mathematics and Artificial Intelligence*, pages 282–296. Springer-Verlag, 1993.

[9] Guozhu Dong and Louxin Zhang. Separating auxiliary arity hierarchy of first-order incremental evaluation using (3k+1)-ary input relations. Technical report, International Journal of Foundations of Computer Science, 1997.

[10] Heinz-Dieter Ebbinghaus and Jörg Flum. *Finite Model Theory*. Springer Verlag, 2005.

[11] Erich Grädel et al. *Finite Model Theory and its Applications*. Springer-Verlag, 2007.

[12] Kousha Etessami. Dynamic tree isomorphism via first-order updates to a relational database. In *Proceedings of PODS 98*, pages 235–243, 1998.

[13] Kousha Etessami and Neil Immerman. Tree canonization and transitive closure. In *Proceedings of LICS 95*, pages 331–341, 1995.

[14] William Hesse. The dynamic complexity of transitive closure is in Dyn-TC^0. In *Proceedings of ICDT 2002*, pages 234–247, 2002.

[15] William Hesse and Neil Immerman. Complete problems for dynamic complexity classes. In *Proceedings of LICS 2002*, pages 313–322, 2002.

[16] Neil Immerman. Languages that capture complexity classes. *SIAM Journal of Computing*, 16:760–778, 1987.

[17] Neil Immerman. Expressibility and parallel complexity. *SIAM J. of Comput*, 18:625–638, 1989.

[18] Leonid Libkin. *Elements of Finite Model Theory*. Springer Verlag, 2004.

[19] Peter B. Miltersen, Jeffresy S. Vitter, Sairam Subramanian, and Roberto Tamassia. Complexity models for incremental computation. *Theoretical*

Computer Science, 130:203–236, 1994.

[20] Sushant Patnaik and Neil Immerman. Dyn-FO: A parallel, dynamic complexity class. *Journal of Computer and System Sciences*, pages 210–221, 1994.

[21] Volker Weber and Thomas Schwentick. Dynamic complexity theory revisited. In *Proceedings of STACS 2005, LNCS 3404*, pages 256–268. Springer, 2005.