On the Existential Fragments of Local First-Order Logics with Data

Benedikt Bollig
CNRS, LMF, ENS Paris-Saclay
Université Paris-Saclay, France

Arnaud Sangnier
IRIF, Université Paris Cité
CNRS,France

Olivier Stietel
CNRS, LMF, ENS Paris-Saclay
Université Paris-Saclay, France
IRIF, Université Paris Cité
CNRS,France

We study first-order logic over unordered structures whose elements carry a finite number of data values from an infinite domain which can be compared wrt. equality. As the satisfiability problem for this logic is undecidable in general, in a previous work, we have introduced a family of local fragments that restrict quantification to neighbourhoods of a given reference point. We provide here the precise complexity characterisation of the satisfiability problem for the existential fragments of this local logic depending on the number of data values carried by each element and the radius of the considered neighbourhoods.

1 Introduction

First-order data logic has emerged to specify properties involving infinite data domains. Potential applications include XML reasoning and the specification of concurrent systems and distributed algorithms. The idea is to extend classic mathematical structures by a mapping that associates with every element of the universe a value from an infinite domain. When comparing data values only for equality, this view is equivalent to extending the underlying signature by a binary relation symbol whose interpretation is restricted to an equivalence relation.

Data logics over word and tree structures were studied in [1, 2]. In particular, the authors showed that two-variable first-order logic on words has a decidable satisfiability problem. Other types of data logics allow *two* data values to be associated with an element [12, 13], though they do not assume a linearly ordered or tree-like universe. Again, satisfiability turned out to be decidable for the two-variable fragment of first-order logic. Other notable extensions, either to multiple data values or to totally ordered data domains, include [5, 11, 15, 17].

When considering an arbitrary number of first-order variables, which we do in this paper, the decidability frontier is quickly crossed without further constraints as soon as the number of allowed data in gretar then two [10]. One of the restrictions we consider here is locality, an essential concept in first-order logic. It is well known that first-order logic is only able to express local properties: a first-order formula can always be written as a combination of properties of elements that have limited, i.e., bounded by a given radius, distance from some reference points [8, 9]. In the presence of (several) data values, imposing a corresponding locality restriction on a logic can help ensuring decidability of its satisfiability problem.

In previous work, we considered a local fragment of first-order data logic over structures whose elements (i) are unordered (as opposed to, e.g., words or trees), and (ii) each carries two data values. We showed that the fragment has a decidable satisfiability problem when restricting local properties to radius 1, while it is undecidable for any radius greater than 1.

D. Della Monica and P. Ganty (Eds.): 13th International Symposium on Games, Automata, Logics and Formal Verification (GandALF 22)
EPTCS 370, 2022, pp. 1–16, doi:10.4204/EPTCS.370.1

In the present paper, we study orthogonal local fragments where global quantification is restricted to being existential (while quantification inside a local property is still unrestricted). We obtain decidability for (i) radius 1 and an arbitrary number of data values, and for (ii) radius 2 and two data values. In all cases, we provide tight complexity upper and lower bounds. Moreover, these results mark the exact decidability frontier: satisfiability is undecidable as soon as we consider radius 3 in presence of two data values, or radius 2 together with three data values.

To give a possible application domain of our logic, consider distributed algorithms running on a cloud of processes. Those algorithms are usually designed to be correct independently of the number of processes executing them. Every process gets some inputs and produces some outputs, usually from an infinite domain. These may include process identifiers, nonces, etc. Inputs and outputs together determine the behavior of a distributed algorithm. A simple example is leader election, where every process gets a unique id, whereas the output should be the id of the elected leader and so be the same for all processes. To formalize correctness properties and to define the intended input-output relation, it is hence essential to have suitable data logics at hand.

Outline. The paper is structured as follows. In Section 2, we recall important notions such as structures and first-order logic, and we introduce the local fragments considered in this paper. Section 3 presents the decidable cases, whereas, in Section 4, we show that all remaining cases lead to undecidability.

This work was partly supported by the project ANR FREDDA (ANR-17-CE40-0013).

2 Structures and first-order logic

2.1 Data Structures

We define here the class of models we are interested in. It consists of sets of nodes containing data values with the assumption that each node is labeled by a set of predicates and carries the same number of values. We consider hence Σ a finite set of unary relation symbols (sometimes called unary predicates) and an integer $D \geq 0$. A *D-data structure* over Σ is a tuple $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma}, f_1, \ldots, f_D)$ (in the following, we simply write $(A, (P_\sigma), f_1, \ldots, f_D)$) where A is a nonempty finite set, $P_\sigma \subseteq A$ for all $\sigma \in \Sigma$, and f_is are mappings $A \to \mathbb{N}$. Intuitively A represents the set of nodes and $f_i(a)$ is the i-th data value carried by a for each node $a \in A$. For $X \subseteq A$, we let $Val_\mathfrak{A}(X) = \{f_i(a) \mid a \in X, i \in \{1,\ldots,D\}\}$. The set of all D-data structures over Σ is denoted by $\mathrm{Data}[\Sigma, D]$.

While this representation is often very convenient to represent data values, a more standard way of representing mathematical structures is in terms of binary relations. For every $(i,j) \in \{1,\ldots,D\} \times \{1,\ldots,D\}$, the mappings $f_1, \ldots, f_D$ determine a binary relation ${}_i\sim^\mathfrak{A}_j \subseteq A \times A$ as follows: $a\ {}_i\sim^\mathfrak{A}_j\ b$ iff $f_i(a) = f_j(b)$. We may omit the superscript $\mathfrak{A}$ if it is clear from the context and if $D = 1$, as there will be only one relation, we way may write $\sim$ for ${}_1\sim_1$.

2.2 First-Order Logic

Let $\mathcal{V} = \{x, y, \ldots\}$ be a countably infinite set of variables. The set $\mathrm{dFO}[\Sigma, D]$ of first-order formulas interpreted over D-data structures over Σ is inductively given by the grammar $\varphi ::= \sigma(x) \mid x\ {}_i\sim_j\ y \mid x = y \mid \varphi \vee \varphi \mid \neg\varphi \mid \exists x.\varphi$, where x and y range over $\mathcal{V}$, σ ranges over Σ, and $i, j \in \{1,\ldots,D\}$. We use standard abbreviations such as $\wedge$ for conjunction and $\Rightarrow$ for implication. We write $\varphi(x_1, \ldots, x_k)$ to indicate that the free variables of φ are among $x_1, \ldots, x_k$. We call φ a *sentence* if it does not contain free variables.

For $\mathfrak{A} = (A, (P_\sigma), f_1, \ldots, f_D) \in \mathrm{Data}[\Sigma, D]$ and a formula $\varphi \in \mathrm{dFO}[\Sigma, D]$, the satisfaction relation $\mathfrak{A} \models_I \varphi$ is defined wrt. an interpretation function $I : \mathscr{V} \to A$. The purpose of I is to assign an interpretation to every (free) variable of φ so that φ can be assigned a truth value. For $x \in \mathscr{V}$ and $a \in A$, the interpretation function $I[x/a]$ maps x to a and coincides with I on all other variables. We then define:

$$\begin{array}{ll} \mathfrak{A} \models_I \sigma(x) \text{ if } I(x) \in P_\sigma & \mathfrak{A} \models_I \varphi_1 \vee \varphi_2 \text{ if } \mathfrak{A} \models_I \varphi_1 \text{ or } \mathfrak{A} \models_I \varphi_2 \\ \mathfrak{A} \models_I x \ {}_i\!\sim_j y \text{ if } I(x) \ {}_i\!\sim^{\mathfrak{A}}_j I(y) & \mathfrak{A} \models_I \neg\varphi \text{ if } \mathfrak{A} \not\models_I \varphi \\ \mathfrak{A} \models_I x = y \text{ if } I(x) = I(y) & \mathfrak{A} \models_I \exists x.\varphi \text{ if there is } a \in A \text{ s.t. } \mathfrak{A} \models_{I[x/a]} \varphi \end{array}$$

Finally, for a data structure $\mathfrak{A} = (A, (P_\sigma), f_1, \ldots, f_D)$, a formula $\varphi(x_1, \ldots, x_k)$ and $a_1, \ldots, a_k \in A$, we write $\mathfrak{A} \models \varphi(a_1, \ldots a_k)$ if there exists an interpretation function I such that $\mathfrak{A} \models_{I[x_1/a_1]\ldots[x_k/a_k]} \varphi$. In particular, for a sentence φ, we write $\mathfrak{A} \models \varphi$ if there exists an interpretation function I such that $\mathfrak{A} \models_I \varphi$.

Example 1 *Assume a unary predicate* $\mathrm{leader} \in \Sigma$. *The following formula from* $\mathrm{dFO}[\Sigma, 2]$ *expresses correctness of a leader-election algorithm: (i) there is a unique process that has been elected leader, and (ii) all processes agree, in terms of their output values (their second data), on the identity (the first data) of the leader:* $\exists x.(\mathrm{leader}(x) \wedge \forall y.\big(\mathrm{leader}(y) \Rightarrow y = x\big)) \wedge \forall y.\exists x.(\mathrm{leader}(x) \wedge x \ {}_1\!\sim_2 y)$.

We are interested here in the satisfiability problem for these logics. Let $\mathscr{F}$ denote a generic class of first-order formulas, parameterized by Σ and D. In particular, for $\mathscr{F} = \mathrm{dFO}$, we have that $\mathscr{F}[\Sigma, D]$ is the class $\mathrm{dFO}[\Sigma, D]$. The satisfiability problem for $\mathscr{F}$ wrt. D-data structures is defined as follows:

DataSat($\mathscr{F}, D$)	
Input:	A finite set Σ and a sentence $\varphi \in \mathscr{F}[\Sigma, D]$.
Question:	Is there $\mathfrak{A} \in \mathrm{Data}[\Sigma, D]$ such that $\mathfrak{A} \models \varphi$?

The following negative result (see [10, Theorem 1]) calls for restrictions of the general logic.

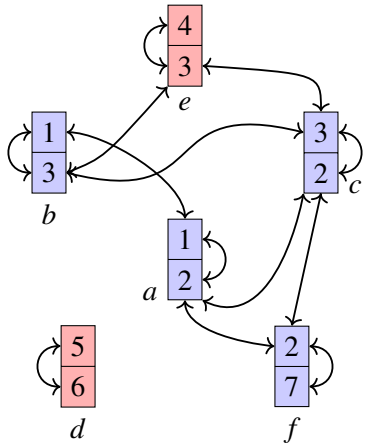
Theorem 1 *[10] The problem* $\mathrm{DataSat}(\mathrm{dFO}, 2)$ *is undecidable, even when we require that* $\Sigma = \emptyset$ *and we do not use* ${}_1\!\sim_2$ *and* ${}_2\!\sim_1$ *in the considered formulas.*

2.3 Local First-Order Logic and its existential fragment

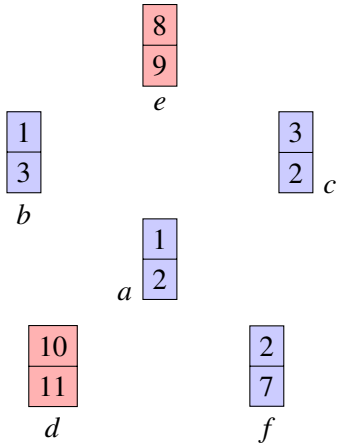
We are interested in logics combining the advantages of $\mathrm{dFO}[\Sigma, D]$, while preserving decidability. With this in mind, we have introduced in [3], for the case of two data values, a *local* restriction, where the scope of quantification in the presence of free variables is restricted to the view of a given element. We present now the defintion of such restrictions adapted to the case of many data values.

First, the view of a node a includes all elements whose distance to a is bounded by a given radius. It is formalized using the notion of a Gaifman graph (for an introduction, see [14]). We use here a variant that is suitable for our setting and that we call *data graph*. Given a data structure $\mathfrak{A} = (A, (P_\sigma), f_1, \ldots, f_D) \in \mathrm{Data}[\Sigma, D]$, we define its *data graph* $\mathscr{G}(\mathfrak{A}) = (V_{\mathscr{G}(\mathfrak{A})}, E_{\mathscr{G}(\mathfrak{A})})$ with set of vertices $V_{\mathscr{G}(\mathfrak{A})} = A \times \{1, \ldots, D\}$ and set of edges $E_{\mathscr{G}(\mathfrak{A})} = \{((a,i),(b,j)) \in V_{\mathscr{G}(\mathfrak{A})} \times V_{\mathscr{G}(\mathfrak{A})} \mid a = b \text{ or } a \ {}_i\!\sim_j b\}$. Figure 1a provides an example of the graph $\mathscr{G}(\mathfrak{A})$ for a data structure with 2 data values.

We then define the distance $d^{\mathfrak{A}}((a,i),(b,j)) \in \mathbb{N} \cup \{\infty\}$ between two elements (a,i) and (b,j) from $A \times \{1, \ldots, D\}$ as the length of the shortest path from (a,i) to (b,j) in $\mathscr{G}(\mathfrak{A})$. For $a \in A$ and $r \in \mathbb{N}$, the *radius-r-ball around a* is the set $B^{\mathfrak{A}}_r(a) = \{(b,j) \in V_{\mathscr{G}(\mathfrak{A})} \mid d^{\mathfrak{A}}((a,i),(b,j)) \leq r \text{ for some } i \in \{1, \ldots, D\}\}$. This ball contains the elements of $V_{\mathscr{G}(\mathfrak{A})}$ that can be reached from $(a,1), \ldots, (a,D)$ through a path of length at most r. On Figure 1a the blue nodes represent $B^{\mathfrak{A}}_2(a)$.



(a) A data structure $\mathfrak{A}$ and $\mathscr{G}(\mathfrak{A})$.



(b) $\mathfrak{A}|_a^2$: the 2 view of a

Figure 1

We now define the r-view of an element a in the D-data structure $\mathfrak{A}$. Intuitively it is a D-data structure with the same elements as $\mathfrak{A}$ but where the data values which are not in the radius-r-ball around a are changed with new values all different one from each other. Let $f_{\text{new}} : A \times \{1,\dots,D\} \to \mathbb{N} \setminus Val_{\mathfrak{A}}(A)$ be an injective mapping. The *r-view of a in* $\mathfrak{A}$ is the structure $\mathfrak{A}|_a^r = (A, (P_\sigma), f'_1, \dots, f'_n) \in \text{Data}[\Sigma, D]$ where its universe is the same as the one of $\mathfrak{A}$ and the unary predicates stay the same and $f'_i(b) = f_i(b)$ if $(b,i) \in B_r^{\mathfrak{A}}(a)$, and $f'_i(b) = f_{\text{new}}((b,i))$ otherwise. On Figure 1b, the structure $\mathfrak{A}|_a^2$ is depicted where the values of the red nodes, not belonging to $B_2^{\mathfrak{A}}(a)$ have been replaced by fresh values not in $\{1,\dots,7\}$.

We are now ready to present the logic r-Loc-dFO$[\Sigma, D]$, where $r \in \mathbb{N}$, interpreted over structures from Data$[\Sigma, D]$. It is given by the grammar

$$\varphi \ ::= \ \langle\!\langle \psi \rangle\!\rangle_x^r \mid x = y \mid \exists x.\varphi \mid \varphi \vee \varphi \mid \neg\varphi$$

where ψ is a formula from dFO$[\Sigma, D]$ with (at most) one free variable x. This logic uses the *local modality* $\langle\!\langle \psi \rangle\!\rangle_x^r$ to specify that the formula ψ should be interpreted over the r-view of the element associated to the variable x. For $\mathfrak{A} \in \text{Data}[\Sigma, D]$ and an interpretation function I, we have indeed $\mathfrak{A} \models_I \langle\!\langle \psi \rangle\!\rangle_x^r$ iff $\mathfrak{A}|_{I(x)}^r \models_I \psi$.

Example 2 *We now illustrate what can be specified by formulas in the logic* 1-Loc-dFO$[\Sigma, 2]$. *We can rewrite the formula from Example 1 so that it falls into our fragment as follows:* $\exists x.(\langle\!\langle \text{leader}(x) \rangle\!\rangle_x^1 \wedge \forall y.(\langle\!\langle \text{leader}(y) \rangle\!\rangle_y^1 \Rightarrow x = y)) \wedge \forall y.\langle\!\langle \exists x.\text{leader}(x) \wedge y \, {}_2{\sim}_1 \, x \rangle\!\rangle_y^1$. *The next formula specifies an algorithm in which all processes suggest a value and then choose a new value among those that have been suggested at least twice:* $\forall x.\langle\!\langle \exists y.\exists z.y \neq z \wedge x \, {}_2{\sim}_1 \, y \wedge x \, {}_2{\sim}_1 \, z \rangle\!\rangle_x^1$. *We can also specify partial renaming, i.e., two output values agree only if their input values are the same:* $\forall x.\langle\!\langle \forall y.(x \, {}_2{\sim}_2 \, y \Rightarrow x \, {}_1{\sim}_1 \, y \rangle\!\rangle_x^1$. *Conversely, the formula* $\forall x.\langle\!\langle \forall y.(x \, {}_1{\sim}_1 \, y \Rightarrow x \, {}_2{\sim}_2 \, y \rangle\!\rangle_x^1$ *specifies partial fusion of equivalences classes.*

In [3], we have studied the decidability status of the satisfiability problem for r-Loc-dFO$[\Sigma, 2]$ with $r \geq 1$ and we have shown that DataSat(2-Loc-dFO, 2) is undecidable and that DataSat(1-Loc-dFO, 2) is decidable when restricting the formulas (and the view of elements) to binary relations belonging to the set $\{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\}$. Whether DataSat(1-Loc-dFO, 2) in its full generality is decidable or not remains an open problem.

We wish to study here the existential fragment of r-Loc-dFO$[\Sigma, D]$ (with $r \geq 1$ and $D \geq 1$) and establish when its satisfiability problem is decidable. This fragment, denoted by $\exists$-r-Loc-dFO$[\Sigma, D]$, is given by the grammar

$$\varphi \ ::= \ \langle\!\langle \psi \rangle\!\rangle_x^r \mid x = y \mid \neg(x = y) \mid \exists x.\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi$$

where ψ is a formula from dFO$[\Sigma, D]$ with (at most) one free variable x. The quantifier free fragment qf-r-Loc-dFO$[\Sigma, D]$ is defined by the grammar $\varphi \ ::= \ \langle\!\langle \psi \rangle\!\rangle_x^r \mid x = y \mid \neg(x = y) \mid \varphi \vee \varphi \mid \varphi \wedge \varphi$.

Remark 1 *Note that for both these fragments, we do not impose any restrictions on the use of quantifiers in the formula ψ located under the local modality $\langle\!\langle \psi \rangle\!\rangle_x^r$.*

3 Decidability results

We show here decidability of DataSat$(\exists$-2-Loc-dFO$, 2)$ and, for all $D \geq 0$, DataSat$(\exists$-1-Loc-dFO$, D)$.

3.1 Preliminary results: 0 and 1 data values

We introduce two preliminary results we shall use in this section to obtain new decidability results. First, note that formulas in dFO$[\Sigma, 0]$ (i.e. where no data is considered) correspond to first order logic formulas with a set of predicates and equality test as a unique relation. As mentioned in Chapter 6.2.1 of [4], these formulas belong to the *Löwenheim class with equality* also called as the relational monadic formulas, and their satisfiability problem is in NEXP. Furthermore, thanks to [6] (Theorem 11), we know that this latter problem is NEXP-hard even if one considers formulas which use only two variables.

Theorem 2 DataSat$(\text{dFO}, 0)$ *is* NEXP*-complete.*

In [16], the authors study the satisfiability problem for Hybrid logic over Kripke structures where the transition relation is an equivalence relation, and they show that it is N2EXP-complete. Furthermore in [7], it is shown that Hybrid logic can be translated to first-order logic in polynomial time and this holds as well for the converse translation. Since 1-data structures can be interpreted as Kripke structures with one equivalence relation, altogether this allows us to obtain the following preliminary result about the satisfiability problem of dFO$[\Sigma, 1]$.

Theorem 3 DataSat$(\text{dFO}, 1)$ *is* N2EXP*-complete.*

3.2 Two data values and balls of radius 2

In this section, we prove that the satisfiability problem for the existential fragment of local first-order logic with two data values and balls of radius two is decidable. To obtain this result we provide a reduction to the satisfiability problem for first-order logic over 1-data structures. Our reduction is based on the following intuition. Consider a 2-data structure $\mathfrak{A} = (A, (P_\sigma), f_1, f_2) \in \text{Data}[\Sigma, 2]$ and an element $a \in A$. If we take an element b in $B_2^{\mathfrak{A}}(a)$, the radius-2-ball around a, we know that either $f_1(b)$ or $f_2(b)$ is a common value with a. In fact, if b is at distance 1 of a, this holds by definition and if b is at distance 2 then b shares an element with c at distance 1 of a and this element has to be shared with a as well so b ends to be at distance 1 of a. The trick consists then in using extra-labels for elements sharing a value with a that can be forgotten and to keep only the value of b not present in a, this construction leading to a 1-data structure. It remains to show that we can ensure that a 1-data structure is the fruit of this

construction in a formula of dFO$[\Sigma',1]$ (where Σ' is obtained from Σ by adding extra predicates).

The first step for our reduction consists in providing a characterisation for the elements located in the radius-1-ball and the radius-2-ball around another element.

Lemma 1 *Let $\mathfrak{A} = (A,(P_\sigma),f_1,f_2) \in \text{Data}[\Sigma,2]$ and $a,b \in A$ and $j \in \{1,2\}$. We have:*

1. *$(b,j) \in B_1^{\mathfrak{A}}(a)$ iff there is $i \in \{1,2\}$ such that $a\ {}_i\!\sim_j^{\mathfrak{A}} b$.*
2. *$(b,j) \in B_2^{\mathfrak{A}}(a)$ iff there exists $i,k \in \{1,2\}$ such that $a\ {}_i\!\sim_k^{\mathfrak{A}} b$.*

Proof: We show both statements:

1. Since $(b,j) \in B_1^{\mathfrak{A}}(a)$, by definition we have either $b = a$ and in that case $a\ {}_j\!\sim_j^{\mathfrak{A}} b$ holds, or $b \neq a$ and necessarily there exists $i \in \{1,2\}$ such that $a\ {}_i\!\sim_j^{\mathfrak{A}} b$.
2. First, if there exists $i,k \in \{1,2\}$ such that $a\ {}_i\!\sim_k^{\mathfrak{A}} b$, then $(b,k) \in B_1^{\mathfrak{A}}(a)$ and $(b,j) \in B_2^{\mathfrak{A}}(a)$ by definition. Assume now that $(b,j) \in B_2^{\mathfrak{A}}(a)$. Hence there exists $i \in \{1,2\}$ such that $d^{\mathfrak{A}}((a,i),(b,j)) \leq 2$. We perform a case analysis on the value of $d^{\mathfrak{A}}((a,i),(b,j))$.
   - **Case** $d^{\mathfrak{A}}((a,i),(b,j)) = 0$. In that case $a = b$ and $i = j$ and we have $a\ {}_i\!\sim_i^{\mathfrak{A}} b$.
   - **Case** $d^{\mathfrak{A}}((a,i),(b,j)) = 1$. In that case, $((a,i),(b,j))$ is an edge in the data graph $\mathscr{G}(\mathfrak{A})$ of $\mathfrak{A}$ which means that $a\ {}_i\!\sim_j^{\mathfrak{A}} b$ holds.
   - **Case** $d^{\mathfrak{A}}((a,i),(b,j)) = 2$. Note that we have by definition $a \neq b$. Furthermore, in that case, there is $(c,k) \in A \times \{1,2\}$ such that $((a,i),(c,k))$ and $((c,k),(b,j))$ are edges in $\mathscr{G}(\mathfrak{A})$. If $c \neq a$ and $c \neq b$, this implies that $a\ {}_i\!\sim_k^{\mathfrak{A}} c$ and $c\ {}_k\!\sim_j^{\mathfrak{A}} b$, so $a\ {}_i\!\sim_j^{\mathfrak{A}} b$ and $d^{\mathfrak{A}}((a,i),(b,j)) = 1$ which is a contradiction. If $c = a$ and $c \neq b$, this implies that $a\ {}_k\!\sim_j^{\mathfrak{A}} b$. If $c \neq a$ and $c = b$, this implies that $a\ {}_i\!\sim_k^{\mathfrak{A}} b$.

□

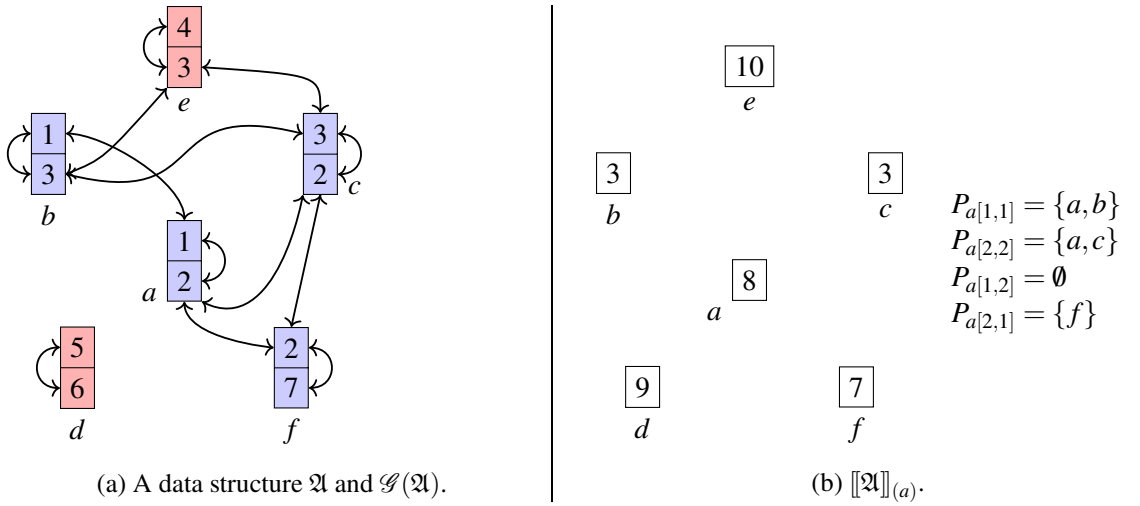
We consider a formula $\varphi = \exists x_1 \ldots \exists x_n.\varphi_{qf}(x_1,\ldots,x_n)$ of $\exists$-2-Loc-dFO$[\Sigma,2]$ in prenex normal form, i.e., such that $\varphi_{qf}(x_1,\ldots,x_n) \in$ qf-2-Loc-dFO$[\Sigma,2]$. We know that there is a structure $\mathfrak{A} = (A,(P_\sigma)_{\sigma\in\Sigma}, f_1,f_2)$ in Data$[\Sigma,2]$ such that $\mathfrak{A} \models \varphi$ if and only if there are $a_1,\ldots,a_n \in A$ such that $\mathfrak{A} \models \varphi_{qf}(a_1,\ldots,a_n)$.

Let $\mathfrak{A} = (A,(P_\sigma)_{\sigma\in\Sigma},f_1,f_2)$ be a structure in Data$[\Sigma,2]$ and a tuple $\vec{a} = (a_1,\ldots,a_n)$ of elements in A^n. We shall present the construction of a 1-data structure $[\![\mathfrak{A}]\!]_{\vec{a}}$ in Data$[\Sigma',1]$ (with $\Sigma \subseteq \Sigma'$) with the same set of nodes as $\mathfrak{A}$, but where each node carries a single data value. In order to retrieve the data relations that hold in $\mathfrak{A}$ while reasoning over $[\![\mathfrak{A}]\!]_{\vec{a}}$, we introduce extra-predicates in Σ' to establish whether a node shares a common value with one of the nodes among $a_1,\ldots,a_n$ in $\mathfrak{A}$.

We now explain formally how we build $[\![\mathfrak{A}]\!]_{\vec{a}}$. Let $\Gamma_n = \{a_p[i,j] \mid p \in \{1,\ldots,n\}, i,j \in \{1,2\}\}$ be a set of new unary predicates and $\Sigma' = \Sigma \cup \Gamma_n$. For every element $b \in A$, the predicates in Γ_n are used to keep track of the relation between the data values of b and the one of $a_1,\ldots,a_n$ in $\mathfrak{A}$. Formally, we define $P_{a_p[i,j]} = \{b \in A \mid \mathfrak{A} \models a_p\ {}_i\!\sim_j b\}$. We now define a data function $f : A \to \mathbb{N}$. We recall for this matter that $Val_{\mathfrak{A}}(\vec{a}) = \{f_1(a_1),f_2(a_1),\ldots,f_1(a_n),f_2(a_n)\}$ and let $f_{\text{new}} : A \to \mathbb{N} \setminus Val_{\mathfrak{A}}(A)$ be an injection. For every $b \in A$, we set:

$$f(b) = \begin{cases} f_2(b) \text{ if } f_1(b) \in Val_{\mathfrak{A}}(\vec{a}) \text{ and } f_2(b) \notin Val_{\mathfrak{A}}(\vec{a}) \\ f_1(b) \text{ if } f_1(b) \notin Val_{\mathfrak{A}}(\vec{a}) \text{ and } f_2(b) \in Val_{\mathfrak{A}}(\vec{a}) \\ f_{\text{new}}(b) \text{ otherwise} \end{cases}$$

Hence depending if $f_1(b)$ or $f_2(b)$ is in $Val_{\mathfrak{A}}(\vec{a})$, it splits the elements of $\mathfrak{A}$ in four categories. If $f_1(b)$ and $f_2(b)$ are in $Val_{\mathfrak{A}}(\vec{a})$, the predicates in Γ_n allow us to retrieve all the data values of b. Given $j \in \{1,2\}$,



(a) A data structure $\mathfrak{A}$ and $\mathscr{G}(\mathfrak{A})$.

(b) $[\![\mathfrak{A}]\!]_{(a)}$.

Figure 2

if $f_j(b)$ is in $Val_{\mathfrak{A}}(\vec{a})$ but $f_{3-j}(b)$ is not, the new predicates will give us the j-th data value of b and we have to keep track of the $(3-j)$-th one, so we save it in $f(b)$. Lastly, if neither $f_1(b)$ nor $f_2(b)$ is in $Val_{\mathfrak{A}}(\vec{a})$, we will never be able to see the data values of b in φ_{q_f} (thanks to Lemma 1), so they do not matter to us. Finally, we have $[\![\mathfrak{A}]\!]_{\vec{a}} = (A, (P_\sigma)_{\sigma\in\Sigma'}, f)$. Figure 2b provides an example of $Val_{\mathfrak{A}}(\vec{a})$ for the data structures depicted on Figure 2a and $\vec{a} = (a)$.

The next lemma formalizes the connection existing between $\mathfrak{A}$ and $[\![\mathfrak{A}]\!]_{\vec{a}}$ with $\vec{a} = (a_1, \ldots, a_n)$.

Lemma 2 *Let $b,c \in A$ and $j,k \in \{1,2\}$ and $p \in \{1,\ldots,n\}$. The following statements then hold.*

1. *If $(b,j) \in B_1^{\mathfrak{A}}(a_p)$ and $(c,k) \in B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$ iff there is $i \in \{1,2\}$ s.t. $b \in P_{a_p[i,j]}$ and $c \in P_{a_p[i,k]}$.*
2. *If $(b,j) \in B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ and $(c,k) \in B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|^2_{a_p} \not\models b\ {}_j{\sim}_k\ c$*
3. *If $(b,j),(c,k) \in B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$ iff either $b\ {}_1{\sim}_1^{[\![\mathfrak{A}]\!]_{\vec{a}}}\ c$ or there exists $p' \in \{1,\ldots,n\}$ and $\ell \in \{1,2\}$ such that $b \in P_{a_{p'}[\ell,j]}$ and $c \in P_{a_{p'}[\ell,k]}$.*
4. *If $(b,j) \notin B_2^{\mathfrak{A}}(a_p)$ and $(c,k) \in B_2^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|^2_{a_p} \not\models b\ {}_j{\sim}_k\ c$*
5. *If $(b,j) \notin B_2^{\mathfrak{A}}(a_p)$ and $(c,k) \notin B_2^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$ iff $b = c$ and $j = k$.*

Proof: We suppose that $\mathfrak{A}|^2_{a_p} = (A, (P_\sigma)_\sigma, f_1^p, f_2^p)$.

1. Assume that $(b,j) \in B_1^{\mathfrak{A}}(a_p)$ and $(c,k) \in B_1^{\mathfrak{A}}(a_p)$. It implies that $f_j^p(b) = f_j(b)$ and $f_k^p(c) = f_k(c)$. Then assume that $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$. As $(b,j) \in B_1^{\mathfrak{A}}(a_p)$, thanks to Lemma 1.1 it means that there is a $i \in \{1,2\}$ such that $a_p\ {}_i{\sim}_j^{\mathfrak{A}}\ b$. So we have $f_k(c) = f_k^p(c) = f_j^p(b) = f_j(b) = f_i(a_p)$, that is $a_p\ {}_i{\sim}_k^{\mathfrak{A}}\ c$. Hence by definition, $b \in P_{a_p[i,j]}$ and $c \in P_{a_p[i,k]}$. Conversely, let $i \in \{1,2\}$ such that $b \in P_{a_p[i,j]}$ and $c \in P_{a_p[i,k]}$. This means that $a_p\ {}_i{\sim}_j^{\mathfrak{A}}\ b$ and $a_p\ {}_i{\sim}_k^{\mathfrak{A}}\ c$. So $f_j^p(b) = f_j(b) = f_i(a_p) = f_k(c) = f_k^p(c)$, that is $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$.
2. Assume that $(b,j) \in B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ and $(c,k) \in B_1^{\mathfrak{A}}(a_p)$. It implies that $f_j^p(b) = f_j(b)$ and $f_k^p(c) = f_k(c)$. Thanks to Lemma 1.1, $(c,k) \in B_1^{\mathfrak{A}}(a_p)$ implies that $f_k(c) \in \{f_1(a_p), f_2(a_p)\}$ and $(b,j) \notin B_1^{\mathfrak{A}}(a_p)$ implies that $f_j(b) \notin \{f_1(a_p), f_2(a_p)\}$. So $\mathfrak{A}|^2_{a_p} \not\models b\ {}_j{\sim}_k\ c$.

3. Assume that $(b,j),(c,k) \in B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$. As previously, we have that $f_j(b) \notin \{f_1(a_p), f_2(a_p)\}$ and $f_k(c) \notin \{f_1(a_p), f_2(a_p)\}$, and thanks to Lemma 1.2, we have $f_{3-j}(b) \in \{f_1(a_p), f_2(a_p)\}$ and $f_{3-k}(b) \in \{f_1(a_p), f_2(a_p)\}$. There is then two cases:
   - Suppose there does not exists $p' \in \{1,\dots,n\}$ such that $f_j(b) \in \{f_1(a_{p'}), f_2(a_{p'})\}$ .This allows us to deduce that $f_j^p(b) = f_j(b) = f(b)$ and $f_k^p(c) = f_k(c)$. If $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$, then necessarily there does not exists $p' \in \{1,\dots,n\}$ such that $f_k(c) \in \{f_1(a_{p'}), f_2(a_{p'})\}$ so we have $f_k^p(c) = f_k(c) = f(c)$ and $f(b) = f(c)$, consequently $b\ {}_1{\sim}_1^{[\![\mathfrak{A}]\!]_{\vec{a}}}\ c$. Similarly assume that $b\ {}_1{\sim}_1^{[\![\mathfrak{A}]\!]_{\vec{a}}}\ c$, this means that $f(b) = f(c)$ and either $b = c$ and $k = j$ or $b \neq c$ and by injectivity of f,we have $f_j(b) = f(b) = f_k(c)$. This allows us to deduce that $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$.
   - If there exists $p' \in \{1,\dots,n\}$ such that $f_j(b) = f_\ell(a_{p'})$ for some $\ell \in \{1,2\}$. Then we have $b \in P_{a_{p'}[\ell,j]}$. Consequently, we have $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$ iff $c \in P_{a_{p'}[\ell,k]}$.
4. We prove the case 4 and 5 at the same time. Assume that $(b,j) \notin B_2^{\mathfrak{A}}(a_p)$. It means that in order to have $f_j^p(b) = f_k^p(c)$, we must have $(b,j) = (c,k)$. So if $(c,k) \in B_2^{\mathfrak{A}}(a_p)$, we can not have $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$ which ends case 4. And if $(c,k) \notin B_2^{\mathfrak{A}}(a_p)$, we have that $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$ iff $b = c$ and $j = k$.

□

We shall now see how we translate the formula $\varphi_{qf}(x_1,\dots,x_n)$ into a formula $[\![\varphi_{qf}]\!](x_1,\dots,x_n)$ in $\mathrm{dFO}[\Sigma',1]$ such that $\mathfrak{A}$ satisfies $\varphi_{qf}(a_1,\dots,a_n)$ if, and only if, $[\![\mathfrak{A}]\!]_{\vec{a}}$ satisfies $[\![\varphi_{qf}]\!](a_1,\dots,a_n)$. Thanks to the previous lemma we know that if $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$ then (b,j) and (c,k) must belong to the same set among $B_1^{\mathfrak{A}}(a_p)$, $B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ and $A \setminus B_2^{\mathfrak{A}}(a_p)$ and we can test in $[\![\mathfrak{A}]\!]_{\vec{a}}$ whether (b,j) is a member of $B_1^{\mathfrak{A}}(a_p)$ or $B_2^{\mathfrak{A}}(a_p)$. Indeed, thanks to Lemmas 1.1 and 1.2, we have $(b,j) \in B_1^{\mathfrak{A}}(a_p)$ iff $b \in \bigcup_{i=1,2} P_{a_p[i,j]}$ and $(b,j) \in B_2^{\mathfrak{A}}(a_p)$ iff $b \in \bigcup_{i=1,2}^{j'=1,2} P_{a_p[i,j']}$. This reasoning leads to the following formulas in $\mathrm{dFO}[\Sigma',1]$ with $p \in \{1,\dots,n\}$ and $j \in \{1,2\}$:

- $\varphi_{j,B_1(a_p)}(y) := a_p[1,j](y) \vee a_p[2,j](y)$ to test if the j-th field of an element belongs to $B_1^{\mathfrak{A}}(a_p)$
- $\varphi_{B_2(a_p)}(y) := \varphi_{1,B_1(a_p)}(y) \vee \varphi_{2,B_1(a_p)}(y)$ to test if a field of an element belongs to $B_2^{\mathfrak{A}}(a_p)$
- $\varphi_{j,B_2(a_p)\setminus B_1(a_p)}(y) := \varphi_{B_2(a_p)}(y) \wedge \neg\varphi_{j,B_1(a_p)}(y)$ to test that the j-th field of an element belongs to $B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$

We shall now present how we use these formulas to translate atomic formulas of the form $y\ {}_j{\sim}_k\ z$ under some $\langle\!\langle - \rangle\!\rangle^2_{x_p}$. For this matter, we rely on the three following formulas of $\mathrm{dFO}[\Sigma',1]$:

- The first formula asks for (y,j) and (z,k) to be in $B_1^1(a_p)$ (where here we abuse notations, using variables for the elements they represent) and for these two data values to coincide with one data value of a_p, it corresponds to Lemma 2.1:

$$\varphi^{r=1}_{j,k,a_p}(y,z) := \varphi_{j,B_1(a_p)}(y) \wedge \varphi_{k,B_1(a_p)}(z) \wedge \bigvee\nolimits_{i=1,2} a_p[i,j](y) \wedge a_p[i,k](z)$$

- The second formula asks for (y,j) and (z,k) to be in $B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ and checks either whether the data values of y and z in $[\![\mathfrak{A}]\!]_{\vec{a}}$ are equal or whether there exist p' and ℓ such that y belongs to $a_{p'}[\ell,j](y)$ and z belongs to $a_{p'}[\ell,k](z)$, it corresponds to Lemma 2.3:

$$\varphi^{r=2}_{j,k,a_p}(y,z) := \varphi_{j,B_2(a_p)\setminus B_1(a_p)}(y) \wedge \varphi_{k,B_2(a_p)\setminus B_1(a_p)}(z) \wedge \big(y \sim z \vee \big(\bigvee\nolimits_{p'=1}^{n} \bigvee\nolimits_{\ell=1}^{2} a_{p'}[\ell,j](y) \wedge a_{p'}[\ell,k](z)\big)\big)$$

- The third formula asks for (y,j) and (z,k) to not belong to $B_2^{\mathfrak{A}}(a_p)$ and for $y=z$, it corresponds to Lemma 2.5:
$$\varphi^{r>2}_{j,k,a_p}(y,z) := \begin{cases} \neg\varphi_{B_2(a_p)}(y) \wedge \neg\varphi_{B_2(a_p)}(z) \wedge y=z & \text{if } j=k \\ \bot & \text{otherwise} \end{cases}$$

Finally, here is the inductive definition of the translation $[\![-]\!]$ which uses sub transformations $[\![-]\!]_{x_p}$ in order to remember the centre of the ball and leads to the construction of $[\![\varphi_{qf}]\!](x_1,\dots,x_n)$:

$$\begin{array}{rcl} [\![\varphi \vee \varphi']\!] & = & [\![\varphi]\!] \vee [\![\varphi']\!] \\ [\![x_p = x'_p]\!] & = & x_p = x'_p \\ [\![\neg\varphi]\!] & = & \neg[\![\varphi]\!] \\ [\![\langle\!\langle \psi \rangle\!\rangle^2_{x_p}]\!] & = & [\![\psi]\!]_{x_p} \\ [\![y\ {}_j{\sim}_k\ z]\!]_{x_p} & = & \varphi^{r=1}_{j,k,a_p}(y,z) \vee \varphi^{r=2}_{j,k,a_p}(y,z) \vee \varphi^{r>2}_{j,k,a_p}(y,z) \\ [\![\sigma(x)]\!]_{x_p} & = & \sigma(x) \\ [\![x=y]\!]_{x_p} & = & x=y \\ [\![\varphi \vee \varphi']\!]_{x_p} & = & [\![\varphi]\!]_{x_p} \vee [\![\varphi']\!]_{x_p} \\ [\![\neg\varphi]\!]_{x_p} & = & \neg[\![\varphi]\!]_{x_p} \\ [\![\exists x.\varphi]\!]_{x_p} & = & \exists x.[\![\varphi]\!]_{x_p} \end{array}$$

Lemma 3 *We have* $\mathfrak{A} \models \varphi_{qf}(\vec{a})$ *iff* $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![\varphi_{qf}]\!](\vec{a})$.

Proof: Because of the inductive definition of $[\![\varphi]\!]$ and that only the atomic formulas $y\ {}_j{\sim}_k\ z$ change, we only have to prove that given $b,c \in A$, we have $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$ iff $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![y\ {}_j{\sim}_k\ z]\!]_{x_p}(b,c)$.

We first suppose that $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$. Using Lemma 2, it implies that (b,j) and (c,k) belong to same set between $B_1^{\mathfrak{A}}(a_p)$, $B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ and $A \setminus B_2^{\mathfrak{A}}(a_p)$. We proceed by a case analysis.

- If $(b,j),(c,k) \in B_1^{\mathfrak{A}}(a_p)$ then by lemma 2.1 we have that $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r=1}_{j,k,a_p}(b,c)$ and thus $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![y\ {}_j{\sim}_k\ z]\!]_{x_p}(b,c)$.
- If $(b,j),(c,k) \in B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ then by lemma 2.3 we have that $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r=2}_{j,k,a_p}(b,c)$ and thus $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![y\ {}_j{\sim}_k\ z]\!]_{x_p}(b,c)$.
- If $(b,j),(c,k) \in A \setminus B_2^{\mathfrak{A}}(a_p)$ then by lemma 2.5 we have that $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r>2}_{j,k,a_p}(b,c)$ and thus $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![y\ {}_j{\sim}_k\ z]\!]_{x_p}(b,c)$.

We now suppose that $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![y\ {}_j{\sim}_k\ z]\!]_{x_p}(b,c)$. It means that $[\![\mathfrak{A}]\!]_{\vec{a}}$ satisfies at least $\varphi^{r=1}_{j,k,a_p}(b,c)$, $\varphi^{r=2}_{j,k,a_p}(b,c)$ or $\varphi^{r>2}_{j,k,a_p}(b,c)$. If $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r=1}_{j,k,a_p}(b,c)$, it implies that (b,j) and (c,k) are in $B_1^{\mathfrak{A}}(a_p)$, and we can then apply lemma 2.1 to deduce that $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$. If $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r=2}_{j,k,a_p}(b,c)$, it implies that (b,j) and (c,k) are in $B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$, and we can then apply lemma 2.3 to deduce that $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$. If $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r>2}_{j,k,a_p}(b,c)$, it implies that (b,j) and (c,k) are in $A \setminus B_2^{\mathfrak{A}}(a_p)$, and we can then apply lemma 2.5 to deduce that $\mathfrak{A}|^2_{a_p} \models b\ {}_j{\sim}_k\ c$. □

To provide a reduction from DATASAT($\exists$-2-Loc-dFO, 2) to DATASAT(dFO, 1), having the formula $[\![\varphi_{qf}]\!](x_1,\dots,x_n)$ is not enough because to use the result of the previous Lemma, we need to ensure that there exists a model $\mathfrak{B}$ and a tuple of elements $(a_1,\dots,a_n)$ such that $\mathfrak{B} \models [\![\varphi_{qf}]\!](a_1,\dots,a_n)$ and as well that there exists $\mathfrak{A} \in \text{Data}[\Sigma,2]$ such that $\mathfrak{B} = [\![\mathfrak{A}]\!]_{\vec{a}}$. We explain now how we can ensure this last point.

Now, we want to characterize the structures of the form $[\![\mathfrak{A}]\!]_{\vec{a}}$. Given $\mathfrak{B} = (A, (P_\sigma)_{\sigma\in\Sigma'}, f) \in \text{Data}[\Sigma',1]$ and $\vec{a} \in A$, we say that $(\mathfrak{B},\vec{a})$ is *well formed* iff there exists a structure $\mathfrak{A} \in \text{Data}[\Sigma,2]$ such that $\mathfrak{B} =$

$[\![\mathfrak{A}]\!]_{\vec{a}}$. Hence $(\mathfrak{B},\vec{a})$ is *well formed* iff there exist two functions $f_1, f_2 : A \to \mathbb{N}$ such that $[\![\mathfrak{A}]\!]_{\vec{a}} = [\![(A,(P_\sigma)_{\sigma\in\Sigma}, f_1, f_2)]\!]_{\vec{a}}$. We state three properties on $(\mathfrak{B},\vec{a})$, and we will show that they characterize being well formed.

1. (Transitivity) For all $b,c \in A$, $p,q \in \{1,\dots,n\}$, $i,j,k,\ell \in \{1,2\}$ if $b \in P_{a_p[i,j]}$, $c \in P_{a_p[i,\ell]}$ and $b \in P_{a_q[k,j]}$ then $c \in P_{a_q[k,\ell]}$.
2. (Reflexivity) For all p and i, we have $a_p \in P_{a_p[i,i]}$
3. (Uniqueness) For all $b \in A$, if $b \in \bigcap_{j=1,2} \bigcup_{p=1,\dots,n}^{i=1,2} P_{a_p[i,j]}$ or $b \notin \bigcup_{j=1,2} \bigcup_{p=1,\dots,n}^{i=1,2} P_{a_p[i,j]}$ then for any $c \in B$ such that $f(c) = f(b)$ we have $c = b$.

Each property can be expressed by a first order logic formula, which we respectively name φ_{tran}, φ_{refl} and φ_{uniq} and we denote by φ_{wf} their conjunction:

$$\begin{array}{ll}
\varphi_{tran} & = \forall y \forall z. \bigwedge_{p,q=1}^{n} \bigwedge_{i,j,k,\ell=1}^{2} \Big(a_p[i,j](y) \wedge a_p[i,\ell](z) \wedge a_q[k,j](y) \Rightarrow a_q[k,\ell](z) \Big) \\
\varphi_{refl}(x_1,\dots,x_n) & = \bigwedge_{p=1}^{n} \bigwedge_{i=1}^{2} a_p[i,i](x_p) \\
\varphi_{uniq} & = \forall y. \Big(\bigwedge_{j=1}^{2} \bigvee_{p=1}^{n} \bigvee_{i=1}^{2} a_p[i,j](y) \vee \bigwedge_{j=1}^{2} \bigwedge_{p=1}^{n} \bigwedge_{i=1}^{2} \neg a_p[i,j](y) \Big) \Rightarrow (\forall z. y \sim z \Rightarrow y = z) \\
\varphi_{wf}(x_1,\dots,x_n) & = \varphi_{tran} \wedge \varphi_{refl}(x_1,\dots,x_n) \wedge \varphi_{uniq}
\end{array}$$

The next lemma expresses that the formula φ_{wf} allows to characterise precisely the 1-data structures in $\mathrm{Data}[\Sigma',1]$ which are well-formed.

Lemma 4 *Let $\mathfrak{B} \in \mathrm{Data}[\Sigma',1]$ and $a_1,\dots,a_n$ elements of $\mathfrak{B}$, then $(\mathfrak{B},\vec{a})$ is well formed iff $\mathfrak{B} \models \varphi_{wf}(\vec{a})$.*

Proof: First, if $(\mathfrak{B},\vec{a})$ is well formed, then there there exists $\mathfrak{A} \in \mathrm{Data}[\Sigma,2]$ such that $\mathfrak{B} = [\![\mathfrak{A}]\!]_{\vec{a}}$ and by construction we have $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi_{wf}(\vec{a})$. We now suppose that $\mathfrak{B} = (A,(P_\sigma)_{\sigma\in\Sigma'}, f)$ and $\mathfrak{B} \models \varphi_{wf}(\vec{a})$. In order to define the functions $f_1, f_2 : A \to \mathbb{N}$, we need to introduce some objects.

We first define a function $g : \{1,\dots,n\} \times \{1,2\} \to \mathbb{N} \setminus Im(f)$ (where $Im(f)$ is the image of f in $\mathfrak{B}$) which verifies the following properties:

- for all $p \in \{1,\dots,n\}$ and $i \in \{1,2\}$, we have $a_p \in P_{a_p[i,3-i]}$ iff $g(p,1) = g(p,2)$;
- for all $p,q \in \{1,\dots,n\}$ and $i,j \in \{1,2\}$, we have $a_q \in P_{a_p[i,j]}$ iff $g(p,i) = g(q,j)$.

We use this function to fix the two data values carried by the elements in $\{a_1,\dots,a_m\}$. We now explain why this function is well founded, it is due to the fact that $\mathfrak{B} \models \varphi_{tran} \wedge \varphi_{refl}(a_1,\dots,a_n)$. In fact, since $\mathfrak{B} \models \varphi_{refl}(a_1,\dots,a_n)$, we have for all $p \in \{1,\dots,n\}$ and $i \in \{1,2\}$, $a_p \in P_{a_p[i,i]}$. Furthermore if $a_p \in P_{a_p[i,j]}$ then $a_p \in P_{a_p[j,i]}$ thanks to the formula φ_{tran}; indeed since we have $a_p \in P_{a_p[i,j]}$ and $a_p \in P_{a_p[i,i]}$ and $a_p \in P_{a_p[j,j]}$, we obtain $a_p \in P_{a_p[j,i]}$. Next, we also have that if $a_q \in P_{a_p[i,j]}$ then $a_p \in P_{a_q[j,i]}$ again thanks to φ_{tran}; indeed since we have $a_q \in P_{a_p[i,j]}$ and $a_p \in P_{a_p[i,i]}$ and $a_q \in P_{a_q[j,j]}$, we obtain $a_p \in P_{a_q[j,i]}$.

We also need a natural d_{out} belonging to $\mathbb{N} \setminus (Im(g) \cup Im(f))$. For $j \in \{1,2\}$, we define f_j as follows for all $b \in A$:

$$f_j(b) = \begin{cases} g(p,i) & \text{if for some } p,i \text{ we have } b \in P_{a_p[i,j]} \\ f(b) & \text{if for all } p,i \text{ we have } b \notin P_{a_p[i,j]} \text{ and for some } p,i \text{ we have } b \in P_{a_p[i,3-j]} \\ d_{out} & \text{if for all } p,i,j' \text{, we have } b \notin P_{a_p[i,j']} \end{cases}$$

Here again, we can show that since $\mathfrak{B} \models \varphi_{tran} \wedge \varphi_{refl}(a_1,\dots,a_n)$, the functions f_1 and f_2 are well founded. Indeed, assume that $b \in P_{a_p[i,j]} \cap P_{a_q[k,j]}$, then we have necessarily that $g(p,i) = g(q,k)$. For this we need to show that $a_p \in a_q[k,i]$ and we use again the formula φ_{tran}. This can be obtained because we have $b \in P_{a_p[i,j]}$ and $a_p \in P_{a_p[i,i]}$ and $b \in P_{a_q[k,j]}$.

We then define $\mathfrak{A}$ as the 2-data-structures $(A,(P_\sigma)_{\sigma\in\Sigma},f_1,f_2)$. It remains to prove that $\mathfrak{B}=[\![\mathfrak{A}]\!]_{\vec{a}}$.

First, note that for all $b\in A$, $p\in\{1,\dots,n\}$ and $i,j\in\{1,2\}$, we have $b\in P_{a_p[i,j]}$ iff $a_p\ {}_i{\sim}{}_j^{\mathfrak{A}}\ b$. Indeed, we have $b\in P_{a_p[i,j]}$, we have that $f_j(b)=g(p,i)$ and since $a_p\in P_{a_p[i,j]}$ we have as well that $f_i(a_p)=g(p,i)$, as a consequence $a_p\ {}_i{\sim}{}_j^{\mathfrak{A}}\ b$. In the other direction, if $a_p\ {}_i{\sim}{}_j^{\mathfrak{A}}\ b$, it means that $f_j(b)=f_i(a_p)=g(p,i)$ and thus $b\in P_{a_p[i,j]}$. Now to have $\mathfrak{B}=[\![\mathfrak{A}]\!]_{\vec{a}}$, one has only to be careful in the choice of function f_{new} while building $[\![\mathfrak{A}]\!]_{\vec{a}}$. We recall that this function is injective and is used to give a value to the elements $b\in A$ such that neither $f_1(b)\in \mathit{Val}_{\mathfrak{A}}(\vec{a})$ and $f_2(b)\notin \mathit{Val}_{\mathfrak{A}}(\vec{a})$ nor $f_1(b)\notin \mathit{Val}_{\mathfrak{A}}(\vec{a})$ and $f_2(b)\in \mathit{Val}_{\mathfrak{A}}(\vec{a})$. For these elements, we make f_{new} matches with the function f and the fact that we define an injection is guaranteed by the formula φ_{uniq}. □

Using the results of Lemma 3 and 4, we deduce that the formula $\varphi=\exists x_1\dots\exists x_n.\varphi_{qf}(x_1,\dots,x_n)$ of $\exists$-2-Loc-dFO$[\Sigma,2]$ is satisfiable iff the formula $\psi=\exists x_1\dots\exists x_n.[\![\varphi_{qf}]\!](x_1,\dots,x_n)\wedge\varphi_{wf}(x_1,\dots,x_n)$ is satisfiable. Note that ψ can be built in polynomial time from φ and that it belongs to dFO$[\Sigma',1]$. Hence, thanks to Theorem 3, we obtain that DataSat$(\exists$-2-Loc-dFO$,2)$ is in N2EXP.

We can as well obtain a matching lower bound thanks to a reduction from DataSat$(\text{dFO},1)$. For this matter we rely on two crucial points. First in the formulas of $\exists$-2-Loc-dFO$[\Sigma,2]$, there is no restriction on the use of quantifiers for the formulas located under the scope of the $\langle\!\langle\cdot\rangle\!\rangle_x^2$ modality and consequently we can write inside this modality a formula of dFO$[\Sigma,1]$ without any modification. Second we can extend a model dFO$[\Sigma,1]$ into a 2-data structure such that all elements and their values are located in the same radius-2-ball by adding everywhere a second data value equal to 0. More formally, let φ be a formula in dFO$[\Sigma,1]$ and consider the formula $\exists x.\langle\!\langle\varphi\rangle\!\rangle_x^2$ where we interpret φ over 2-data structures (this formula simply never mentions the values located in the second fields). We have then the following lemma.

Lemma 5 *There exists* $\mathfrak{A}\in\text{Data}[\Sigma,1]$ *such that* $\mathfrak{A}\models\varphi$ *if and only if there exists* $\mathfrak{B}\in\text{Data}[\Sigma,2]$ *such that* $\mathfrak{B}\models\exists x.\langle\!\langle\varphi\rangle\!\rangle_x^2$.

Proof: Assume that there exists $\mathfrak{A}=(A,(P_\sigma)_{\sigma\in\Sigma},f_1)$ in Data$[\Sigma,1]$ such that $\mathfrak{A}\models\varphi$. Consider the 2-data structure $\mathfrak{B}=(A,(P_\sigma)_{\sigma\in\Sigma},f_1,f_2)$ such that $f_2(a)=0$ for all $a\in A$. Let $a\in A$. It is clear that we have $\mathfrak{B}|_a^2=\mathfrak{B}$ and that $\mathfrak{B}|_a^2\models\varphi$ (because $\mathfrak{A}\models\varphi$ and φ never mentions the second values of the elements since it is a formula in dFO$[\Sigma,1]$). Consequently $\mathfrak{B}\models\exists x.\langle\!\langle\varphi\rangle\!\rangle_x^2$.

Assume now that there exists $\mathfrak{B}=(A,(P_\sigma)_{\sigma\in\Sigma},f_1,f_2)$ in Data$[\Sigma,2]$ such that $\mathfrak{B}\models\exists x.\langle\!\langle\varphi\rangle\!\rangle_x^2$. Hence there exists $a\in A$ such that $\mathfrak{B}|_a^2\models\varphi$, but then by forgetting the second value in $\mathfrak{B}|_a^2$ we obtain a model in Data$[\Sigma,1]$ which satisfies φ. □

Since DataSat$(\text{dFO},1)$ is N2EXP-hard (see Theorem 3), we obtain the desired lower bound.

Theorem 4 *The problem* DataSat$(\exists$-2-Loc-dFO$,2)$ *is* N2EXP*-complete.*

3.3 Balls of radius 1 and any number of data values

Let $D\geq 1$. We first show that DataSat$(\exists$-1-Loc-dFO$,D)$ is in NEXP by providing a reduction towards DataSat$(\text{dFO},0)$. This reduction uses the characterisation of the radius-1-ball provided by Lemma 1 and is very similar to the reduction provided in the previous section. In fact, for an element b located in the radius-1-ball of another element a, we use extra unary predicates to explicit which are the values of b that are common with the values of a. We provide here the main step of this reduction whose proof follows the same line as the one of Theorem 4.

We consider a formula $\varphi=\exists x_1\dots\exists x_n.\varphi_{qf}(x_1,\dots,x_n)$ of $\exists$-1-Loc-dFO$[\Sigma,D]$ in prenex normal form, i.e., such that $\varphi_{qf}(x_1,\dots,x_n)\in$ qf-1-Loc-dFO$[\Sigma,D]$. We know that there is a structure $\mathfrak{A}=(A,(P_\sigma)_{\sigma\in\Sigma},$

$f_1, f_2, \ldots, f_D)$ in Data$[\Sigma, D]$ such that $\mathfrak{A} \models \varphi$ if and only if there are $a_1, \ldots, a_n \in A$ such that $\mathfrak{A} \models \varphi_{qf}(a_1, \ldots, a_n)$. Let then $\mathfrak{A} = (A, (P_\sigma)_{\sigma \in \Sigma}, f_1, f_2, \ldots, f_D)$ in Data$[\Sigma, D]$ and a tuple $\vec{a} = (a_1, \ldots, a_n)$ of elements in A^n. Let $\Omega_n = \{a_p[i,j] \mid p \in \{1, \ldots, n\}, i, j \in \{1, \ldots, D\}\}$ be a set of new unary predicates and $\Sigma' = \Sigma \cup \Omega_n$. For every element $b \in A$, the predicates in Ω_n are used to keep track of the relation between the data values of b and the one of $a_1, \ldots, a_n$ in $\mathfrak{A}$. Formally, we have $P_{a_p[i,j]} = \{b \in A \mid \mathfrak{A} \models a_p \; {}_i\!\sim_j b\}$. Finally, we build the 0-data-structure $[\![\mathfrak{A}]\!]'_{\vec{a}} = (A, (P_\sigma)_{\sigma \in \Sigma'})$. Similarly to Lemma 2, we have the following connection between $\mathfrak{A}$ and $[\![\mathfrak{A}]\!]'_{\vec{a}}$.

Lemma 6 *Let $b, c \in A$ and $j, k \in \{1, \ldots, D\}$ and $p \in \{1, \ldots, n\}$. The following statements hold:*

1. *If $(b, j) \in B_1^{\mathfrak{A}}(a_p)$ and $(c, k) \in B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|^1_{a_p} \models b \; {}_j\!\sim_k c$ iff there is $i \in \{1, 2\}$ s.t. $b \in P_{a_p[i,j]}$ and $c \in P_{a_p[i,k]}$.*
2. *If $(b, j) \notin B_1^{\mathfrak{A}}(a_p)$ and $(c, k) \in B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|^1_{a_p} \not\models b \; {}_j\!\sim_k c$*
3. *If $(b, j) \notin B_1^{\mathfrak{A}}(a_p)$ and $(c, k) \notin B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|^1_{a_p} \models b \; {}_j\!\sim_k c$ iff $b = c$ and $j = k$.*

We shall now see how we translate the formula $\varphi_{qf}(x_1, \ldots, x_n)$ into a formula $[\![\varphi_{qf}]\!]'(x_1, \ldots, x_n)$ in dFO$[\Sigma', 0]$ such that $\mathfrak{A}$ satisfies $\varphi_{qf}(a_1, \ldots, a_n)$ if, and only if, $[\![\mathfrak{A}]\!]'_{\vec{a}}$ satisfies $[\![\varphi_{qf}]\!](a_1, \ldots, a_n)$. As in the previous section, we introduce the following formula in dFO$[\Sigma', 0]$ with $p \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, D\}$ to test if the j-th field of an element belongs to $B_1^{\mathfrak{A}}(a_p)$:

$$\varphi_{j, B_1(a_p)}(y) := \bigvee_{i \in \{1, \ldots, D\}} a_p[i, j](y)$$

We now present how we translate atomic formulas of the form $y \; {}_j\!\sim_k z$ under some $\langle\!\langle - \rangle\!\rangle^1_{x_p}$. For this matter, we rely on two formulas of dFO$[\Sigma', 0]$ which can be described as follows:

- The first formula asks for (y, j) and (z, k) to be in $B_1^1(a_p)$ (here we abuse notations, using variables for the elements they represent) and for these two data values to coincide with one data value of a_p, it corresponds to Lemma 6.1:

$$\psi^{r=1}_{j,k,a_p}(y, z) := \varphi_{j, B_1(a_p)}(y) \wedge \varphi_{k, B_1(a_p)}(z) \wedge \bigvee\nolimits_{i=1}^{D} a_p[i, j](y) \wedge a_p[i, k](z)$$

- The second formula asks for (y, j) and (z, k) to not belong to $B_1^{\mathfrak{A}}(a_p)$ and for $y = z$, it corresponds to Lemma 6.3:

$$\psi^{r>1}_{j,k,a_p}(y, z) := \begin{cases} \bigwedge_{i=1}^{D} (\neg \varphi_{i, B_1(a_p)}(y) \wedge \neg \varphi_{i, B_1(a_p)}(z)) \wedge y = z & \text{if } j = k \\ \bot & \text{otherwise} \end{cases}$$

Finally, as before we provide an inductive definition of the translation $[\![-]\!]'$ which uses subtransformations $[\![-]\!]'_{x_p}$ in order to remember the centre of the ball and leads to the construction of $[\![\varphi_{qf}]\!]'(x_1, \ldots, x_n)$. We only detail the case

$$[\![y \; {}_j\!\sim_k z]\!]'_{x_p} = \psi^{r=1}_{j,k,a_p}(y, z) \vee \psi^{r>1}_{j,k,a_p}(y, z)$$

as the other cases are identical as for the translation $[\![-]\!]$ shown in the previous section. This leads to the following lemma (which is the pendant of Lemma 3).

Lemma 7 *We have $\mathfrak{A} \models \varphi_{qf}(\vec{a})$ iff $[\![\mathfrak{A}]\!]'_{\vec{a}} \models [\![\varphi_{qf}]\!]'(\vec{a})$.*

As we had to characterise the well-formed 1-data structure, a similar trick is necessary here. For this matter, we use the following formulas:

$$\begin{array}{ll}\psi_{tran} & = \forall y \forall z. \bigwedge_{p,q=1}^{n} \bigwedge_{i,j,k,\ell=1}^{D} \Big(a_p[i,j](y) \wedge a_p[i,\ell](z) \wedge a_q[k,j](y) \Rightarrow a_q[k,\ell](z) \Big) \\ \psi_{refl}(x_1,\ldots,x_n) & = \bigwedge_{p=1}^{n} \bigwedge_{i=1}^{D} a_p[i,i](x_p) \\ \psi_{wf}(x_1,\ldots,x_n) & = \psi_{tran} \wedge \psi_{refl}(x_1,\ldots,x_n)\end{array}$$

Finally with the same reasoning as the one given in the previous section, we can show that the formula $\varphi = \exists x_1 \ldots \exists x_n . \varphi_{qf}(x_1,\ldots,x_n)$ of $\exists$-1-Loc-dFO$[\Sigma, D]$ is satisfiable iff the formula $\exists x_1 \ldots \exists x_n . [\![\varphi_{qf}]\!]'(x_1,\ldots,x_n) \wedge \psi_{wf}(x_1,\ldots,x_n)$ is satisfiable. Note that this latter formula can be built in polynomial time from φ and that it belongs to dFO$[\Sigma', 0]$. Hence, thanks to Theorem 2, we obtain that DATASAT($\exists$-1-Loc-dFO$, D$) is in NEXP. The matching lower bound is as well obtained the same way by reducing DATASAT(dFO$, 0$) to DATASAT($\exists$-1-Loc-dFO$, D$) showing that a formula φ in dFO$[\Sigma, 0]$ is satisfiable iff the formula $\exists x . \langle\!\langle \varphi \rangle\!\rangle_x^1$ in $\exists$-D-Loc-dFO$[\Sigma, 1]$ is satisfiable.

Theorem 5 *For all $D \geq 1$, the problem* DATASAT($\exists$-1-Loc-dFO$, D$) *is* NEXP*-complete.*

4 Undecidability results

We show here DATASAT($\exists$-3-Loc-dFO$, 2$) and DATASAT($\exists$-2-Loc-dFO$, 3$) are undecidable. To obtain this we provide reductions from DATASAT(dFO$, 2$) and we use the fact that any 2-data structure can be interpreted as a radius-3-ball of a 2-data structure or respectively as a radius-2-ball of a 3-data structure.

4.1 Radius 3 and two data values

In order to reduce DATASAT(dFO$, 2$) to DATASAT($\exists$-3-Loc-dFO$, 2$), we show that we can transform slightly any 2-data structure $\mathfrak{A}$ into an other 2-data structure $\mathfrak{A}_{\mathsf{ge}}$ such that $\mathfrak{A}_{\mathsf{ge}}$ corresponds to the radius-3-ball of any element of $\mathfrak{A}_{\mathsf{ge}}$ and this transformation has some kind of inverse. Furthermore, given a formula $\varphi \in$ dFO$[\Sigma, 2]$, we transform it into a formula $T(\varphi)$ in $\exists$-3-Loc-dFO$[\Sigma', 2]$ such that $\mathfrak{A}$ satisfies φ iff $\mathfrak{A}_{\mathsf{ge}}$ satisfies $T(\varphi)$. What follows is the formalisation of this reasoning.

Let $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, f_2)$ be a 2-data structure in Data$[\Sigma, 2]$ and ge be a fresh unary predicate not in Σ. From $\mathfrak{A}$ we build the following 2-data structure $\mathfrak{A}_{\mathsf{ge}} = (A', (P'_\sigma)_\sigma, f'_1, f'_2) \in$ Data$[\Sigma \cup \{\mathsf{ge}\}, 2]$ such that:

- $A' = A \uplus Val_{\mathfrak{A}}(A) \times Val_{\mathfrak{A}}(A)$,
- for $i \in \{1,2\}$ and $a \in A$, $f'_i(a) = f_i(a)$ and for $(d_1,d_2) \in Val_{\mathfrak{A}}(A) \times Val_{\mathfrak{A}}(A)$, $f_i((d_1,d_2)) = d_i$,
- for $\sigma \in \Sigma$, $P'_\sigma = P_\sigma$,
- $P_{\mathsf{ge}} = Val_{\mathfrak{A}}(A) \times Val_{\mathfrak{A}}(A)$.

Hence to build $\mathfrak{A}_{\mathsf{ge}}$ from $\mathfrak{A}$ we have added to the elements of $\mathfrak{A}$ all pairs of data presented in $\mathfrak{A}$ and in order to recognise these new elements in the structure we use the new unary predicate ge. We add these extra elements to ensure that all the elements of the structure are located in the radius-3-ball of any element of $\mathfrak{A}_{\mathsf{ge}}$. We have then the following property.

Lemma 8 $\mathfrak{A}_{\mathsf{ge}}|_a^3 = \mathfrak{A}_{\mathsf{ge}}$ *for all $a \in A'$.*

Proof: Let $b \in A'$ and $i,j \in \{1,2\}$. We show that $d^{\mathfrak{A}_{\mathsf{ge}}}((a,i),(b,j)) \leq 3$. i.e. that there is a path of length at most 3 from (a,i) to (b,j) in the data graph $\mathscr{G}(\mathfrak{A}_{\mathsf{ge}})$. By construction of $\mathfrak{A}_{\mathsf{ge}}$, there is an element $c \in A'$ such that $f_1(c) = f_i(a)$ and $f_2(c) = f_j(b)$. So we have the path $(a,i),(c,1),(c,2),(b,j)$ of length at most 3 from (a,i) to (b,j) in $\mathscr{G}(\mathfrak{A}_{\mathsf{ge}})$. □

Conversely, to $\mathfrak{A} = (A,(P_\sigma)_\sigma,f_1,f_2) \in \mathrm{Data}[\Sigma \cup \{\mathsf{ge}\},2]$, we associate $\mathfrak{A}_{\setminus \mathsf{ge}} = (A',(P'_\sigma)_\sigma,f'_1,f'_2) \in \mathrm{Data}[\Sigma,2]$ where:

- $A' = A \setminus P_{\mathsf{ge}}$,
- for $i \in \{1,2\}$ and $a \in A'$, $f'_i(a) = f_i(a)$,
- for $\sigma \in \Sigma$, $P'_\sigma = P'_\sigma \setminus P_{\mathsf{ge}}$.

Finally we inductively translate any formula $\varphi \in \mathrm{dFO}[\Sigma,2]$ into $T(\varphi) \in \mathrm{dFO}[\Sigma \cup \{\mathsf{ge}\},2]$ by making it quantify over elements not labeled with ge: $T(\sigma(x)) = \sigma(x)$, $T(x \sim_j y) = x \sim_j y$, $T(x=y) = (x=y)$, $T(\exists x.\varphi) = \exists x.\neg\mathsf{ge}(x) \wedge T(\varphi)$, $T(\varphi \vee \varphi') = T(\varphi) \vee T(\varphi')$ and $T(\neg\varphi) = \neg T(\varphi)$.

Lemma 9 *Let φ be a sentence in* $\mathrm{dFO}[\Sigma,2]$, $\mathfrak{A} \in \mathrm{Data}[\Sigma,2]$ *and* $\mathfrak{B} \in \mathrm{Data}[\Sigma \cup \{\mathsf{ge}\},2]$. *The two following properties hold:*

- $\mathfrak{A} \models \varphi$ *iff* $\mathfrak{A}_{\mathsf{ge}} \models T(\varphi)$
- $\mathfrak{B}_{\setminus \mathsf{ge}} \models \varphi$ *iff* $\mathfrak{B} \models T(\varphi)$.

Proof: As for any $\mathfrak{A} \in \mathrm{Data}[\Sigma,2]$ we have $(\mathfrak{A}_{\mathsf{ge}})_{\setminus \mathsf{ge}} = \mathfrak{A}$, it is sufficient to prove the second point. We reason by induction on φ. Let $\mathfrak{A} = (A,(P_\sigma)_\sigma,f_1,f_2) \in \mathrm{Data}[\Sigma \cup \{\mathsf{ge}\},2]$ and let $\mathfrak{A}_{\setminus \mathsf{ge}} = (A',(P'_\sigma)_\sigma,f'_1,f'_2) \in \mathrm{Data}[\Sigma,2]$. The inductive hypothesis is that for any formula $\varphi \in \mathrm{dFO}[\Sigma,2]$ (closed or not) and any context interpretation function $I : \mathscr{V} \to A'$ we have $\mathfrak{A}_{\setminus \mathsf{ge}} \models_I \varphi$ iff $\mathfrak{A} \models_I T(\varphi)$. Note that the inductive hypothesis is well founded in the sense that the interpretation I always maps variables to elements of the structures.

We prove two cases: when φ is a unary predicate and when φ starts by an existential quantification, the other cases being similar. First, assume that $\varphi = \sigma(x)$ where $\sigma \in \Sigma$. $\mathfrak{A}_{\setminus \mathsf{ge}} \models_I \sigma(x)$ holds iff $I(x) \in P'_\sigma$. As $I(x) \in A \setminus P_{\mathsf{ge}}$, we have $I(x) \in P'_\sigma$ iff $I(x) \in P_\sigma$, which is equivalent to $\mathfrak{A} \models_I T(\sigma(x))$. Second assume $\varphi = \exists x.\varphi'$. Suppose that $\mathfrak{A}_{\setminus \mathsf{ge}} \models_I \exists x.\varphi'$. Thus, there is a $a \in A'$ such that $\mathfrak{A}_{\setminus \mathsf{ge}} \models_{I[x/a]} \varphi'$. By inductive hypothesis, we have $\mathfrak{A} \models_{I[x/a]} T(\varphi')$. As $a \in A' = A \setminus P_{\mathsf{ge}}$, we have $\mathfrak{A} \models_{I[x/a]} \neg\mathsf{ge}(x)$, so $\mathfrak{A} \models_I \exists x.\neg\mathsf{ge}(x) \wedge T(\varphi')$ as desired. Conversely, suppose that $\mathfrak{A} \models_I T(\exists x.\varphi')$. It means that there is a $a \in A$ such that $\mathfrak{A} \models_{I[x/a]} \neg\mathsf{ge}(x) \wedge T(\varphi')$. So we have that $a \in A' = A \setminus P_{\mathsf{ge}}$, which means that $I[x/a]$ takes values in A and we can apply the inductive hypothesis to get that $\mathfrak{A}_{\setminus \mathsf{ge}} \models_{I[x/a]} \varphi'$. So we have $\mathfrak{A}_{\setminus \mathsf{ge}} \models_I \exists x.\varphi'$. □

From Theorem 1, we know that $\textsc{DataSat}(\mathrm{dFO},2)$ is undecidable. From a closed formula $\varphi \in \mathrm{dFO}[\Sigma,2]$, we build the formula $\exists x.\langle\!\langle T(\varphi)\rangle\!\rangle^3_x \in \exists\text{-3-Loc-dFO}[\Sigma \cup \{\mathsf{ge}\},2]$. Now if φ is satisfiable, it means that there exists $\mathfrak{A} \in \mathrm{Data}[\Sigma,2]$ such that $\mathfrak{A} \models \varphi$. By Lemma 9, $\mathfrak{A}_{\mathsf{ge}} \models T(\varphi)$. Let a be an element of $\mathfrak{A}$, then thanks to Lemma 8, we have $\mathfrak{A}_{\mathsf{ge}}|^3_a \models T(\varphi)$. Finally by definition of our logic, $\mathfrak{A}_{\mathsf{ge}} \models \exists x.\langle\!\langle T(\varphi)\rangle\!\rangle^3_x$. So $\exists x.\langle\!\langle T(\varphi)\rangle\!\rangle^3_x$ is satisfiable. Now assume that $\exists x.\langle\!\langle T(\varphi)\rangle\!\rangle^3_x$ is satisfiable. So there exist $\mathfrak{A} \in \mathrm{Data}[\Sigma \cup \{\mathsf{ge}\},2]$ and an element a of $\mathfrak{A}$ such that $\mathfrak{A}|^3_a \models T(\varphi)$. Using Lemma 9, we obtain $(\mathfrak{A}|^3_a)_{\setminus \mathsf{ge}} \models \varphi$. Hence φ is satisfiable. This shows that we can reduce $\textsc{DataSat}(\mathrm{dFO},2)$ to $\textsc{DataSat}(\exists\text{-3-Loc-dFO},2)$.

Theorem 6 *The problem* $\textsc{DataSat}(\exists\text{-3-Loc-dFO},2)$ *is undecidable.*

4.2 Radius 2 and three data values

We provide here a reduction from DATASAT(dFO,2) to DATASAT(∃-2-Loc-dFO,3). The idea is similar to the one used in the proof of Lemma 5 to show that DATASAT(∃-2-Loc-dFO,2) is N2EXP-hard by reducing DATASAT(dFO,1). Indeed we have the following Lemma.

Lemma 10 *Let φ be a formula in* dFO$[\Sigma,2]$. *There exists $\mathfrak{A} \in$ Data$[\Sigma,2]$ such that $\mathfrak{A} \models \varphi$ if and only if there exists $\mathfrak{B} \in$ Data$[\Sigma,3]$ such that $\mathfrak{B} \models \exists x.\langle\!\langle \varphi \rangle\!\rangle^2_x$.*

Proof: Assume that there exists $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2)$ in Data$[\Sigma,2]$ such that $\mathfrak{A} \models \varphi$.Consider the 3-data structure $\mathfrak{B} = (A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2, f_3)$ such that $f_3(a) = 0$ for all $a \in A$. Let $a \in A$. It is clear that we have $\mathfrak{B}|^2_a = \mathfrak{B}$ and that $\mathfrak{B}|^2_a \models \varphi$ (because $\mathfrak{A} \models \varphi$ and φ never mentions the third values of the elements since it is a formula in dFO$[\Sigma,1]$). Consequently $\mathfrak{B} \models \exists x.\langle\!\langle \varphi \rangle\!\rangle^2_x$.

Assume now that there exists $\mathfrak{B} = (A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2, f_3)$ in Data$[\Sigma,3]$ such that $\mathfrak{B} \models \exists x.\langle\!\langle \varphi \rangle\!\rangle^2_x$. Hence there exists $a \in A$ such that $\mathfrak{B}|^2_a \models \varphi$, but then by forgetting the third value in $\mathfrak{B}|^2_a$ we obtain a model in Data$[\Sigma,3]$ which satisfies φ. □

Using Theorem 1, we obtain the following result.

Theorem 7 *The problem* DATASAT(∃-2-Loc-dFO,3) *is undecidable.*

References

[1] M. Bojanczyk, C. David, A. Muscholl, T. Schwentick & L. Segoufin (2011): *Two-variable logic on data words*. *ACM Trans. Comput. Log.* 12(4), pp. 27:1–27:26, doi:10.1145/1970398.1970403.

[2] M. Bojanczyk, A. Muscholl, T. Schwentick & L. Segoufin (2009): *Two-variable logic on data trees and XML reasoning*. *J. ACM* 56(3), doi:10.1145/1516512.1516515.

[3] Benedikt Bollig, Arnaud Sangnier & Olivier Stietel (2021): *Local First-Order Logic with Two Data Values*. In: *FSTTCS'21*, *LIPIcs* 213, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 39:1–39:15, doi:10.4230/LIPIcs.FSTTCS.2021.39.

[4] Egon Börger, Erich Grädel & Yuri Gurevich (1997): *The Classical Decision Problem*. Perspectives in Mathematical Logic, Springer, doi:10.1023/A:1008334715902.

[5] N. Decker, P. Habermehl, M. Leucker & D. Thoma (2014): *Ordered Navigation on Multi-attributed Data Words*. In Paolo Baldan & Daniele Gorla, editors: *CONCUR'14*, *Lecture Notes in Computer Science* 8704, Springer, pp. 497–511, doi:10.1007/978-3-662-44584-6_34.

[6] Kousha Etessami, Moshe Y. Vardi & Thomas Wilke (2002): *First-Order Logic with Two Variables and Unary Temporal Logic*. *Inf. Comput.* 179(2), pp. 279–295, doi:10.1006/inco.2001.2953.

[7] Melvin Fitting (2012): *Torben Braüner, Hybrid Logic and its Proof-Theory, Applied Logic Series Volume 37, Springer, 2011, pp. XIII+231. ISBN: 978-94-007-0001-7*. *Stud Logica* 100(5), pp. 1051–1053, doi:10.1007/s11225-012-9439-2.

[8] H. Gaifman (1982): *On local and nonlocal properties*. In J. Stern, editor: *Logic Colloquium '81*, North-Holland, pp. 105–135, doi:10.1016/S0049-237X(08)71879-2.

[9] W. Hanf (1965): *Model-theoretic methods in the study of elementary logic*. In J.W. Addison, L. Henkin & A. Tarski, editors: *The Theory of Models*, North Holland, pp. 132–145, doi:10.2307/2271017.

[10] A. Janiczak (1953): *Undecidability of some simple formalized theories*. *Fundamenta Mathematicae* 40, pp. 131–139, doi:10.2307/2964197.

[11] A. Kara, T. Schwentick & T. Zeume (2010): *Temporal Logics on Words with Multiple Data Values*. In Kamal Lodaya & Meena Mahajan, editors: *FSTTCS'10*, *LIPIcs* 8, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 481–492, doi:10.4230/LIPIcs.FSTTCS.2010.481.

[12] E. Kieronski (2005): *Results on the Guarded Fragment with Equivalence or Transitive Relations*. In C.-H. Luke Ong, editor: *CSL'05*, *Lecture Notes in Computer Science* 3634, Springer, pp. 309–324, doi:10.1007/11538363_22.

[13] E. Kieronski & L. Tendera (2009): *On Finite Satisfiability of Two-Variable First-Order Logic with Equivalence Relations*. In: *LICS'09*, IEEE, pp. 123–132, doi:10.1109/LICS.2009.39.

[14] L. Libkin (2004): *Elements of Finite Model Theory*. Texts in Theoretical Computer Science. An EATCS Series, Springer, doi:10.1007/978-3-662-07003-1.

[15] A. Manuel & T. Zeume (2013): *Two-Variable Logic on 2-Dimensional Structures*. In Simona Ronchi Della Rocca, editor: *CSL'13*, *LIPIcs* 23, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 484–499, doi:10.4230/LIPIcs.CSL.2013.484.

[16] Martin Mundhenk & Thomas Schneider (2009): *The Complexity of Hybrid Logics over Equivalence Relations*. *J. Log. Lang. Inf.* 18(4), pp. 493–514, doi:10.1007/s10849-009-9089-6.

[17] T. Tan (2014): *Extending two-variable logic on data trees with order on data values and its automata*. *ACM Trans. Comput. Log.* 15(1), pp. 8:1–8:39, doi:10.1145/2559945.