A Note on C^2 Interpreted over Finite Data-Words

Bartosz Bednarczyk
Computational Logic Group, TU Dresden, Germany
Institute of Computer Science, University of Wrocław, Poland
bartosz.bednarczyk@cs.uni.wroc.pl

Piotr Witkowski
Institute of Computer Science, University of Wrocław, Poland
piotr.witkowski@cs.uni.wroc.pl

Abstract

We consider the satisfiability problem for the two-variable fragment of first-order logic extended with counting quantifiers, interpreted over finite words with data, denoted here with $C^2[\leq, succ, \sim, \pi_{bin}]$. In our scenario, we allow for using arbitrary many uninterpreted binary predicates from π_{bin}, two navigational predicates $\leq$ and *succ* over word positions as well as a data-equality predicate $\sim$. We prove that the obtained logic is undecidable, which contrasts with the decidability of the logic without counting by Montanari, Pazzaglia and Sala [27]. We supplement our results with decidability for several sub-fragments of $C^2[\leq, succ, \sim, \pi_{bin}]$, *e.g.* without binary predicates, without successor *succ*, or under the assumption that the total number of positions carrying the same data value in a data-word is bounded by an a priori given constant.

2012 ACM Subject Classification Theory of computation → Logic and verification

Keywords and phrases Two-variable logic, data-words, VASS, decidability, undecidability, counting

Digital Object Identifier 10.4230/LIPIcs.TIME.2020.17

Funding *Bartosz Bednarczyk*: supported by "Diamentowy Grant" no. DI2017 006447.
Piotr Witkowski: supported by NCN grant no. 2016/21/B/ST6/01444.

1 Introduction

Finite data-words [8], *i.e.* finite words, where each position carries letters from a finite alphabet as well as a data value from some countably-infinite data domain, are ubiquitous in formal verification. They can be used to describe executions of array-accessing programs [1], runs of counter machines [18], outputs of timed systems [9] or database transaction logs [28]. However, reasoning about them is not simple: the main obstacle is the unboundedness of the data domain. We discuss some of the recently proposed approaches to solve the problem.

The first solution is stemming from automata theory. To deal with data-words, the notion of class automata [5, 3], data automata [4], register automata [22] or session automata [7] were proposed. Usually, these are automata equipped with a set of registers, used to store the current data value in the memory. Of course, such registers must be suited to store information of unknown size and must be properly restrained: one can easily fall into a trap that the proposed automata model can simulate zero tests, which usually causes undecidability [26]. Unfortunately, proposed automata models lack good algorithmic properties. By way of example, the emptiness problem for class memory automata is equivalent to reachability in vector-addition systems and hence, non-elementary [16]. Moreover, the model of class automata is not closed under complementation, which results in an undecidable equivalence problem. Some weaker subclasses of class automata were considered *e.g.* in [15].

Thus, in this paper, we rather focus on declarative models like logics. Being aware of the plethora of different automata models proposed in the past, it is not hard to conclude that a similar situation should occur for logics. The most famous frameworks, tailored to reason about data-words, are temporal logics and fragments of first-order logics. The former

ones were well-developed in the recent years: *e.g.* LTL with freeze quantifier, which can be used as a logical counterpart of a register, was proposed in [19]. Other examples are the temporal logic of repeating values [18], the PathLog [21] and LTL data quantification [32], just to mention a few of them. As far as the existential monadic second-order logic [6] and first-order logic are considered [4], the logics were rather neglected, probably due to their high complexity or even undecidability. The logics generally allows for quantification over words' positions, to compare elements with navigational predicates and to check whether data values of two elements coincide by means of data-equality predicate. The logics FO or EMSO are immediately undecidable. The only known decidable fragments are the two-variable fragments: $\mathrm{FO}^2[\mathit{succ}, \sim]$, $\mathrm{FO}^2[\leq, \sim]$ and $\mathrm{FO}^2[\leq, \mathit{succ}, \sim]$, where $\leq$ is a linear order over words' positions, *succ* is its induced successor relation and $\sim$ is a data-equality predicate. The first two logics are known to be NExpTime-complete [28, 4], while the last one is known to be interreducible to the reachability problem in vector addition systems with states. Our work will focus on extending FO^2 to make the logic more expressive yet decidable.

We encourage the reader to check the latest surveys on the topic [17, 13] or PhD theses [24, 28, 14] to improve his understanding of the state-of-the-art of the problem and to get a glimpse of the maze of data languages.

1.1 Our motivation

We aim at extending the framework of the two-variable logic FO^2 on data-words to the realm of quantitative properties. Our goal is very modest: we would like to understand the behaviour of FO^2 under the extensions of counting quantifiers. Such quantifiers can be used to express basic quantitative properties like: "*there are at least five data repartitions in the run of the machine*" or "*each request has exactly one corresponding grant with the same data value*". The techniques dealing with counting quantifiers were well-developed in the least 10 years, see *e.g.* [29, 30, 12, 11], hence there is a hope that they work well also in the context of data-word reasoning. We hope our work will lay the foundation on an expressive specification language for data-words involving an interplay between counting capabilities and data values.

1.2 Our contribution

We study satisfiability problems for $\mathrm{C}^2[\leq, \mathit{succ}, \sim, \pi_{bin}]$, *i.e.* the two-variable logic with counting quantifiers admitting a linear order predicate $\leq$, its induced successor relation *succ*, a data-equality predicate $\sim$ and a set of uninterpreted binary symbols π_{bin}. Our results are:

- In Section 3 we show that $\mathrm{C}^2[\leq, \mathit{succ}, \sim, \pi_{bin}]$ is undecidable, in sharp contrast to the logic without counting [27]. The proof reuses ideas from [2] on how to encode runs of Minsky Machines on data-words. The key property is the existence of C^2 formula imposing that a fresh binary relation is a one-to-one matching of domain elements, whilst being a refinement of $\sim$. We also discuss how the undecidability result transfers to similar logics, *e.g.* to $\mathrm{C}^2[\leq, \sim, \pi_{bin}]$. Negative results are supplemented by several decidability results.
- In Section 4 we show that both $\mathrm{C}^2[\mathit{succ}, \sim, \pi_{bin}]$ and $\mathrm{C}^2[\leq, \sim]$ logics are NExpTime-complete. The NExpTime lower-bound is trivially inherited from FO^2, but the upper bounds are less trivial. For the former logic, we provide a reduction to the appropriate logic on data-trees [11], for which the NExpTime-completeness was recently shown by the second author and his colleagues. For the latter logic, namely, for $\mathrm{C}^2[\leq, \sim]$, we show that any satisfiable formula has a model with only exponentially many equivalence classes. Such a property allows us to replace the data-equality tests with equi-satisfiability of polynomially many unary predicates, which encodes the class number in binary. Finally, the tight NExpTime upper bound is obtained by employing as a black-box an algorithm from [12] for deciding finite satisfiability for the logic on words without data values.

- In Section 5 we deal with the finite satisfiability of $C^2[\leq, succ, \sim]$. We employ a counting-quantifier elimination technique to get rid of seemingly more expressive concepts from the logic. The logic $C^2[\leq, succ, \sim]$ turned out to be VASS-complete, *i.e.* complete for the class of all problems elementarily reducible to the reachability in Vector Addition Systems (solvable in ACKERMANN time [25] with a non-elementary TOWER lower bound [16]).
- Finally, in the last section, we establish the most technically challenging result of this paper, namely the VASS-completeness of $C^2[\leq, succ, \sim, \pi_{bin}]$ under the restriction that each equivalence class has a uniform bound k on their sizes. Differently phrased, it means that a single data value can occur in a data-word only, a priori given, constant number of times. In those logics, we allow for using data-equality predicate with $\sim_k$ instead of the full data-equality $\sim$. To solve the satisfiability problem, we propose a translation from $C^2[\leq, succ, \sim_k, \pi_{bin}]$ to $C^2[\leq, succ, \pi_{bin}]$, that is the logic without $\sim_k$. The main problem is that transitivity is not expressible with only two variables, and hence we cannot hope for an "easy" translation. To achieve our goal we take an input formula φ and link it with some formulae imposing a colouring of the structure with some fresh letters smartly encoding information to which class given elements belong.

2 Preliminaries

Let Σ be a finite *alphabet* (*i.e.* a set of unary predicates) and let $\mathbb{D}$ be a countably-infinite *data domain*. A *data word* is an element from $(2^\Sigma \times \mathbb{D})^*$. A *language* is a set of data words. In our setting, we are interested in fragments of first-order logic describing data-words. We agree that the formulae have direct access to the alphabet Σ, allowing to use the letters as unary predicates. To the contrary, the data-values from $\mathbb{D}$ are stored implicitly: the only allowed operation is a comparison of data-values between positions with an equivalence relation $\sim$ called the *data equality predicate*. In the paper, we follow the usual notations [4].

2.1 Logics

The two-variable[1] logic $FO^2[\leq, succ, \sim]$ interpreted over finite data-words is a fragment of first-order logic featuring only two variables x, y and equipped with a vocabulary of arbitrary many unary predicates (aka letters), two *navigational predicates* over the words' positions, namely a linear order $\leq$ and its induced successor relation *succ*, and $\sim$. Whenever $x \leq y$ holds, we say that x is to the left of y. Additionally, we extend the logic with an arbitrarily large set of uninterpreted binary predicates ${\pi_{bin}}^2$, forming the logic $FO^2[\leq, succ, \sim, \pi_{bin}]$. In this paper, we mostly work with counting extensions of FO^2, denoted here with C^2. Such logics extend the previous ones with the so-called counting quantifiers $\exists^{\geq k}, \exists^{\leq k}$, with their natural meaning, *i.e.* $\exists^{\geq k} x.\varphi$ is satisfied in a data-word w if at least k positions, when substituted as x, satisfy φ.

We are interested in the *finite satisfiability problem* phrased as "*given a formula φ is there a data word satisfying φ?*". The current state-of-the-art of the problem is presented in the table below. All of the claimed bounds are tight and the appropriate reference is cited (with [H] we indicate that the result is shown in this paper).[3]

[1] With σ in $\mathcal{L}[\sigma]$ we indicate what kinds of binary relations can be in the logic.
[2] They can be used with counting quantifiers *e.g.* to express Presburger constraints over universes [31].
[3] Recall that VASS complexity class is composed of all problems elementarily reducible to VASS-reachability.

Table 1 The complexity of the satisfiability problem for FO^2 and C^2 over finite data words. All stated complexity bounds are tight.

	$[], [\leq, succ]$	$[succ, \sim, \pi_{bin}]$	$[\leq, \sim]$	$[\leq, succ, \sim]$	$[\leq, \sim, \pi_{bin}]$	$[\leq, succ, \sim, \pi_{bin}]$
FO^2	NEXP [20]	NEXP [28]	NEXP [4]	VASS [4]	NEXP [27]	VASS [27]
C^2	NEXP [12]	NEXP [11]	NEXP [H]	VASS [H]	Undecidable [H]	

2.2 Normal Forms

It is usually very convenient to work with the formulae in tailored normal forms. In the paper we will present two of them. Reducing a formula into such forms is usually simple and requires well-known techniques, *cf.* [23, 29]. Hence, routine proofs are omitted.

We employ two types of *Scott-normal forms* for C^2, the latter being tailored especially for construction in Section 5. In the remaining sections we employ weak normal forms. Their main advantage is that they are computable in polynomial time *cf.* [12].

$$\varphi = \forall x \forall y\ \chi \wedge \bigwedge_{i=1}^{n} \forall x \exists^{\bowtie_i C_i} y\ \chi_i, \tag{1}$$

with $\bowtie_i \in \{\leq, \geq\}$, quantifier-free χ, χ_i and with all C_i being natural numbers. A *1-type* is a maximal consistent set of literals over Σ involving only the variable x. Note that the number of 1-types is over Σ is exponential in the size of Σ. Likewise, a *2-type* is a maximal consistent set of literals over Σ involving only the variables x and y and containing the literal $x \neq y$. In Section 5 we use the following normal form.

$$\varphi = \forall x \forall y\ \alpha \wedge \bigwedge_{i=1}^{n} \forall x\ \left(\pi_i(x) \to \exists^{\bowtie_i C_i} y\ \beta_i\right) \wedge \bigwedge_{i=1}^{n'} \forall x\ \left(\pi'_i(x) \to \exists^{\bowtie_i C'_i} y\ \gamma_i\right), \tag{2}$$

where α is quantifier-free formula, $\bowtie_i \in \{\leq, =, \geq\}$, π_i, π'_i are 1-types and β_i, γ_i are 2-types and each β_i contains $x \sim y$ and each γ_i contains $x \not\sim y$. Its main feature is the presence of 1-types and 2-types, *i.e.* since each element has a unique 1-type, the types and location of its witnesses y are given explicitly in the 2-types β_i.

3 Undecidability of the full logic

For a moment we move to a slightly more general framework, namely, we assume that each position of a data word carries a pair of data (d_1, d_2) from a product of two countably infinite sets $\mathbb{D}_1$ and $\mathbb{D}_2$, rather than just a single datum. In this scenario, we allow to use two equivalence relations $\sim_1$ and $\sim_2$, responsible, respectively, for the data tests of first and of the second coordinate. It is known that even the most natural logic for this setting, namely $\mathrm{FO}^2[\leq, succ, \sim_1, \sim_2]$, becomes immediately undecidable [4]. Moreover, the FO^2 logic remains undecidable even when the second datum is treated as a refinement of the first one, *i.e.* when a formula $\forall x \forall y\ x \sim_2 y \to x \sim_1 y$ is a tautology [2]. Here we explain how to modify the undecidability proof from [2, Appendix A.1] to infer undecidability of $\mathrm{C}^2[\leq, succ, \sim, \pi_{bin}]$.

To prove undecidability of $\mathrm{FO}^2[\leq, succ, \sim_1, \sim_2]$ (under the proviso that $\sim_2$ is a refinement of $\sim_1$), the authors of [2] provided a reduction from the halting problem for Minsky Machines [26]. They encoded successful runs of a machine as data words from $\mathcal{L}$, where:

$$\mathcal{L} = s_1 s_2 (i_1 + i_2 + d_1 + d_2 + e_1 s_1 + e_2 s_2)^* e_2 e_1.$$

An intuition behind such language is fairly simple: the letters i_k and d_k correspond to the incrementation and the decrementation of the k-th counter, while the letters s_k and e_k

correspond to zero tests. Then the subwords composed of all positions between each s_k and e_k are assumed to have the equal first datum, *i.e.* are in the same $\sim_1$ equivalence class. As the next step, the relation $\sim_2$ was employed to match each incrementation i_k with an appropriate d_k from the same $\sim_1$-class. Finally, consistency between two neighbouring configurations was handled with a two-variable formula without any data-equality predicates.

Note that the equivalence relation $\sim_2$ was only used to match occurrences of i_k with occurrences of d_k and vice-versa. The same property can be stated with a single one-to-one binary relation required to be a subset of $\sim_1$. And such a property is easily expressible in C^2:

$$\forall x\,(\forall y\; x \sim_2 y \to x \sim y) \wedge \forall x\,\left(\exists^{\leq 1} y\; x \sim_2 y \wedge \exists^{\leq 1} y\; y \sim_2 x\right)$$

With such an interpretation of $\sim_2$, the undecidability proof of [2] can be read without any changes as an undecidability proof for $\mathrm{C}^2[\leq, \mathit{succ}, \sim, \pi_{bin}]$. Thus we conclude:

▶ **Theorem 1.** *Satisfiability of* $\mathrm{C}^2[\leq, \mathit{succ}, \sim, \pi_{bin}]$ *over finite data-words is undecidable, even if* π_{bin} *contains only a single binary relation and the only allowed counting quantifier is* $\exists^{\leq 1}$.

Note that in the presence of uninterpreted binary symbols in the language, the successor relation *succ* can be defined in $\mathrm{C}^2[\leq, \sim, \pi_{bin}]$ cf. [12, Lemma 3.1]. Hence we can also infer the undecidability of the logic without the successor relation.

▶ **Theorem 2.** *Satisfiability of* $\mathrm{C}^2[\leq, \sim, \pi_{bin}]$ *over finite data-words is undecidable.*

4 When only one navigational binary relation is allowed

As a first step towards decidability, we consider sublogics of $\mathrm{C}^2[\leq, \mathit{succ}, \sim, \pi_{bin}]$ without uninterpreted binary symbols π_{bin} and with only a single binary navigational predicate.

For the case when only the *succ* relation is allowed, we reuse a recent result on C^2 interpreted over trees with data. It was shown in [11] that the logic $\mathrm{C}^2[\downarrow, \sim, \pi_{bin}]$, namely C^2 with two distinguished relations interpreted, respectively, as a parent-child relation in a tree and as an equivalence relation is NExpTime-complete. Note that a word can be seen as a tree, where each node has at most one child. Moreover, by employing the formula $\forall x \exists^{\leq 1} y\; x \downarrow y$ we can enforce that the intended tree models are actually words. Hence from [11] we conclude:

▶ **Theorem 3.** *The satisfiability for* $\mathrm{C}^2[\mathit{succ}, \sim]$ *and* $\mathrm{C}^2[\mathit{succ}, \sim, \pi_{bin}]$ *is NExpTime-complete.*

To obtain a tight NExpTime upper bound for the next logic, namely for $\mathrm{C}^2[\leq, \sim]$, we closely follow the line of NExpTime-completeness proof for $\mathrm{FO}^2[\leq, \sim]$ from [4, Lemma 19].

We first show that any satisfiable $\mathrm{C}^2[\leq, \sim]$ formula φ has a model with at most exponentially many equivalence classes. This is done by taking an arbitrary model and performing some surgery on it. More precisely, we first mark an appropriate number of equivalence classes at the beginning (together with an appropriate number of their elements) as well as on the end. Then, if any non-marked element needs a witness, it should find one in an equivalence class of some marked element. Once such a lemma is shown, we can assign some number to each of the equivalence classes. Since there are only exponentially many of them, their numbers can be encoded with only polynomially many bits represented with only polynomially many fresh unary predicates. Thus in that setting, testing whether two positions carry the same data-value boils down to checking the number of their equivalence classes and it can be handled easily in FO^2. Finally, we rewrite the formula into a $\sim$-free one and use a black-box an NExpTime algorithm for solving $\mathrm{C}^2[\leq]$ from [12]. Now we show:

▶ **Lemma 4.** *Any satisfiable* $\mathrm{C}^2[\leq, \sim]$*-formula* φ *has a model, in which the total number of* $\sim$*-equivalence classes* $\mathit{eq}(\varphi)$ *is bounded exponentially in* $|\varphi|$.

Proof. Assume that φ is in the weak normal form (*cf.* Eq. 1). Let C be the maximal number appearing in the counting quantifiers and let t be the number of all possible 1-types over the vocabulary of φ. Note that both C and t are exponential in $|\varphi|$. In the forthcoming proof, we will show how to construct a model of φ with at most $t \cdot 2(C{+}1)$ different classes.
Let $\mathfrak{A}$ be a model of φ. For each 1-type α we mark the first $C{+}1$ positions of $\mathfrak{A}$ with type α from mutually different classes [or all of them if there are less than $C{+}1$ of them in $\mathfrak{A}$]. Analogously we repeat the process for the last $C{+}1$ positions of type α. Let $\mathfrak{B}$ be a subword of $\mathfrak{A}$ composed of only those positions of $\mathfrak{A}$, which has the same data as some marked element.

We will show that $\mathfrak{B} \models \varphi$. Since the described construction preserves 1-types, we conclude that $\mathfrak{B}$ satisfies the $\forall x \forall y\ \chi$ part of φ (because the satisfaction of χ depends only on 1-types realized in a model). Moreover, the satisfaction of all subformulae of the form $\forall \exists^{\leq C_i}$ are preserved too, due to the fact that $\mathfrak{B}$ is a substructure of $\mathfrak{A}$. The tricky part here is show preservation of satisfaction of $\forall x \exists^{\geq C_i} y\ \chi_i(x,y)$ formulae. Take an arbitrary position p from $\mathfrak{B}$ and consider what kind of witnesses y it has in $\mathfrak{A}$ to satisfy $\chi(x,y)$. All possible y from the same class as p are preserved in the construction, so they can still serve as witnesses for p. It could be also the case that p had k (where $k \leq C$) witnesses from a different class, to the right of p. But since at least k classes were marked during the construction, then p can take as witnesses some k elements from those marked classes (in the worst case such elements coincide with the original ones). For witnesses to the left of p we proceed analogously. Thus, by considering all sub-cases, we infer $\mathfrak{B} \models \varphi$. The total number of different classes in $\mathfrak{B}$ is bounded by $t \cdot 2(C{+}1)$, and hence is only exponential in $|\varphi|$. ◀

Let $p_0, p_1, \ldots, p_m$ be fresh unary predicates, such that $2^{m+1} \geq eq(\varphi) > 2^m$ holds for $eq(\varphi)$ obtained from the above lemma. As we have already mentioned, once the number of equivalence classes is bounded, checking whether two elements x and y are related by $\sim$ boils down to checking whether they encode the same number on p_i predicates. Hence, we can replace all subformulae of the form $x \sim y$ in φ with a formula $\wedge_{i=0}^{m}(p_i(x) \leftrightarrow p_i(y))$. The formulae obtained in this way are (purely) $\mathrm{C}^2[\leq]$ formulae and are of polynomial size. Thus by employing an NExpTime algorithm for deciding fin-sat of $\mathrm{C}^2[\leq]$ from [12] we obtain:

▶ **Theorem 5.** *Satisfiability for* $\mathrm{C}^2[\leq, \sim]$ *over finite data-words is NExpTime-complete.*

5 When uninterpreted relations are disallowed

In this section, we focus on the most expressive variant of data logics without binary predicates, namely on $\mathrm{C}^2[\leq, succ, \sim]$. It is known that its FO^2 version is Vass-complete [4]. Here we show that the Vass-completeness transfers also to its C^2 counterpart, which will be done by a model-preserving translation from $\mathrm{C}^2[\leq, succ, \sim]$ to $\mathrm{FO}^2[\leq, succ, \sim]$. Note that since $\mathrm{FO}^2[\leq, succ, \sim]$ is non-elementary, we do not need to care too much about how complex complexity-wise the reduction will be, as long as its size is bounded by some elementary function. Before we start, we will assume that the input formula is in the Scott-like normal form (2) defined in Section 2.2. Our plan is to gradually remove all $\forall \exists^{\bowtie}$ conjuncts from φ, replacing them with some equisatisfiable formulae without counting quantifiers. Let $C = 1 + \max_{i=1}^{n}\{C_i\}$ and let us proceed as follows. Observe that any $\forall \exists^{\bowtie} \psi$ conjunct requires, for a fixed x, at most C witnesses for its satisfaction. Hence, once we would know in advance how many witnesses for ψ the element x has, we would immediately know whether the $\forall \exists^{\bowtie} \psi$ formula is satisfied or not. Thus, we aim at providing such information. In order to do that, we introduce fresh unary predicates labelling the elements of the model, both globally and locally in every equivalence class, numbering occurrences the certain 1-types (from the start and from the end of the model) up to the threshold C. It will suffice to eliminate the counting.

To explain the technique, let us first consider the case of $\bigwedge_{i=1}^{n} \forall x \ \left(\pi_i(x) \to \exists^{\bowtie_i C_i} y \ \beta_i\right)$ conjuncts, which we prefer to call *class conjuncts*, since they speak about witnesses y from the same equivalence class as x. For each 1-type π and $i \in \{1, 2, \ldots, C+1\}$ we introduce fresh unary predicates $\mathit{cl\text{-}left}_i^\pi$ and $\mathit{cl\text{-}right}_i^\pi$ and we impose their interpretation, *e.g.* that $\mathit{cl\text{-}left}_i^\pi(x)$ holds iff x is the i-th occurrence (counted from 1 from the beginning of the model) of the 1-type π in the equivalence class of x. Writing the formulae imposing such interpretation is easy, *e.g.* to impose that $\mathit{cl\text{-}left}_2^\pi$ means the second occurrence of the type π, we write:

$$\forall x \mathit{cl\text{-}left}_2^\pi(x) \leftrightarrow (\pi(x) \wedge \exists y.(y < x \wedge y \sim x \wedge \pi(y)) \wedge \forall y \, (y < x \wedge y \sim x \wedge \pi(x) \to \mathit{cl\text{-}left}_1^\pi(y)))$$

▶ **Fact 6.** *There is an* $\mathrm{FO}^2[\leq, \mathit{succ}, \sim]$ *formula* φ_{cl} *such that for every model* $\mathfrak{A} \models \varphi_{cl}$ *and every* $1 \leq i \leq C$ *we have that* $\mathit{cl\text{-}left}_i^\pi(x)$ *(resp.* $\mathit{cl\text{-}right}_i^\pi(x)$*) holds iff* x *is the* i*-th occurrence from the beginning of the model (resp. the end) of the* 1*-type* π *in the equivalence class of* x*.*

The above fact allows us to eliminate counting quantifiers from the class conjuncts from φ. By way of example, consider the formula $\pi(x) \to \exists^{\leq C_i} y. \, (\pi'(y) \wedge x \sim y \wedge y < x \wedge \neg \mathit{succ}(x, y))$, which states that each x of the 1-type π should see at most C_i elements of the type π' (in its equivalence class) strictly to its left. By employing Fact 6 we can rewrite it into: $\pi(x) \to \neg \exists y.(y < x \wedge x \sim y \wedge \neg \mathit{succ}(x, y) \wedge \mathit{cl\text{-}left}_{C_i+1}(x))$. Other cases are treated similarly.

▶ **Lemma 7.** *Any* $\mathrm{C}^2[\leq, \mathit{succ}, \sim]$ *formula* φ *in the normal form can be transformed into equisatisfiable* $\mathrm{C}^2[\leq, \mathit{succ}, \sim]$ *formula* φ' *without counting quantifiers in the class conjuncts.*

Now we discuss how to eliminate counting quantifiers in the non-class conjuncts. The method will be similar to the previous one, but the introduced labelling will be more involved. By way of example, consider the formula $\pi(x) \to \exists^{\geq C_i} y. \, (\pi'(y) \wedge x \not\sim y \wedge x < y \wedge \neg \mathit{succ}(y, x))$, which states that each x of the 1-type π requires at least C_i witnesses, outside the equivalence class of x, of the 1-type π' strictly to the right of x. It would be tempting to claim that the global labelling of the last C elements with the 1-type π' would be sufficient for our purposes. Unfortunately, it is not: it could be the case that the last C elements are in the same class. To omit such difficulties, we label up C^2 elements with the type π in total (from the left and from the right) with the predicates $\mathit{gl\text{-}left}_i^\pi$, $\mathit{gl\text{-}right}_i^\pi$, but we require that no more than C elements from the same class is marked (*i.e.* in our numbering we simply skip elements from the class containing C labelled elements). In means that if an element needs to find witnesses from outside of its class, it should find them among the marked elements. Once again, providing such a labelling is an easy exercise in $\mathrm{FO}^2[\leq, \mathit{succ}, \sim]$.

▶ **Fact 8.** *There is an* $\mathrm{FO}^2[\leq, \mathit{succ}, \sim]$ *formula* φ_{gl} *such that for every model* $\mathfrak{A} \models \varphi_{gl}$ *and every* $1 \leq i \leq C^2$ *we have that* $\mathit{gl\text{-}left}_i^\pi(x)$ *(resp.* $\mathit{gl\text{-}right}_i^\pi(x)$*) holds iff* x *is the* i*-th occurrence from the beginning of the model (resp. the end) of the* 1*-type* π*, skipping in the enumeration all the elements already having* C *elements labelled with some* $\mathit{gl\text{-}left}_j^\pi(x)$ *(resp.* $\mathit{gl\text{-}right}_j^\pi(x)$*) in their equivalence class.*

Now we will discuss how to employ such a labelling to eliminate counting quantifiers in the non-class conjuncts. Recall the toy formula: $\pi(x) \to \exists^{\geq C_i} y. \, (\pi'(y) \wedge x \not\sim y \wedge x < y \wedge \neg \mathit{succ}(y, x))$. We need to state that an element x can see at least C elements of the 1-type π' to its right, outside its equivalence class. Observe that we already enumerated elements of the 1-type π' inside the equivalence class of x. Hence if there are j elements of the type π' to the right of x, *i.e.* $\mathit{cl\text{-}right}_j^{\pi'}(y)$ is satisfied for some $y > x$ having the same data-value as x, it suffices to state that x can see to its right an element labelled with $\mathit{gl\text{-}right}_{C_i+j}^{\pi'}$. And this can be defined with an $\mathrm{FO}^2[\leq, \mathit{succ}, \sim]$ formula. By applying analogous reasoning, one can eliminate counting quantifiers also in the other cases. Hence we conclude the following lemma:

▶ **Lemma 9.** *Any* $\mathrm{C}^2[\leq, succ, \sim]$ *formula* φ *in the normal form can be transformed into equisatisfiable* $\mathrm{C}^2[\leq, succ, \sim]$ *formula* φ' *without counting quantifiers in the non-class conjuncts. Moreover,* φ' *does not introduce any counting quantiers in the class conjuncts.*

By employing Lemma 7, Lemma 9 and Vass-completeness of $\mathrm{FO}^2[\leq, succ, \sim]$ we establish the main theorem of this section.

▶ **Theorem 10.** *For any* $\mathrm{C}^2[\leq, succ, \sim]$ *formula* φ *there exists an equisatisfiable* $\mathrm{FO}^2[\leq, succ, \sim]$ *formula* φ' *of an elementary size in* $|\varphi|$ *and hence,* $\mathrm{C}^2[\leq, succ, \sim]$ *is Vass-complete.*

6 C^2 with full linear order and bounded data-tests

In this section we prove that the decidability of the full logic can be regained, under a reasonable assumption that no more than k (for a fixed number k) elements in the model share the same data-value. To express such a restriction in the logical terms, we employ the relation $\sim_k$, interpreted as an equivalence relation with equivalence classes of size at most k. We show that the logic $\mathrm{C}^2[\leq, succ, \sim_k, \pi_{bin}]$ is Vass-complete. The proof goes via a reduction to $\mathrm{C}^2[\leq, succ, \pi_{bin}]$. Since the latter logic is Vass-complete [12] we conclude the result.

More precisely, given a $\mathrm{C}^2[\leq, succ, \sim_k, \pi_{bin}]$ formula φ we will produce an equisatisfiable $\mathrm{C}^2[\leq, succ, \pi_{bin}]$ formula φ_{tr} by adding to φ conjuncts that encode some $\sim_k$ properties and enable model transformations that preserve satisfiability of φ and the interpretations of $\leq$ and $succ$. The essential part of the reduction will be to use these transformations on an arbitrary model of φ_{tr} to produce a model of φ in which $\sim_k$ is interpreted as a bounded equivalence relation. By $\mathcal{W}(\leq, succ, \pi_{bin})$ denote the class of all words and by $\mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$ its subclass where $\sim_k$ is interpreted as described above.

6.1 Plethora of types

We make extensive use of the notions of (atomic) 1- and 2-types. In both cases, we take the notion of consistency to incorporate the constraint that the distinguished predicate $\sim_k$ is interpreted as a reflexive and a symmetric relation (note that transitivity would require three variables and thus cannot be enforced in the same way). If τ is a 2-type, we denote by τ^{-1} the 2-type obtained by exchanging the variables x and y in τ, and call τ^{-1} the *inverse* of τ. We denote by $\mathrm{tp}_1(\tau)$ the 1-type obtained by removing from τ any literals containing y; and we denote by $\mathrm{tp}_2(\tau)$ the 1-type obtained by first removing from τ any literals containing x, and then replacing all occurrences of y by x. Evidently, $\mathrm{tp}_2(\tau) = \mathrm{tp}_1(\tau^{-1})$. We equivocate freely between finite sets of formulae and their conjunctions; thus, we treat 1-types and 2-types as formulae, where convenient. Let $\mathfrak{A}$ be any structure interpreting Σ. If $a \in A$, then there exists a unique 1-type π such that $\mathfrak{A} \models \pi[a]$; we denote π by $\mathrm{tp}^{\mathfrak{A}}[a]$ and say that a *realizes* π. If, in addition, $b \in A \setminus \{a\}$, then there exists a unique 2-type τ such that $\mathfrak{A} \models \tau[a, b]$; we denote τ by $\mathrm{tp}^{\mathfrak{A}}[a, b]$ and say that the pair a, b *realizes* τ. Evidently, in that case, $\tau^{-1} = \mathrm{tp}^{\mathfrak{A}}[b, a]$; $\mathrm{tp}_1(\tau) = \mathrm{tp}^{\mathfrak{A}}[a]$; and $\mathrm{tp}_2(\tau) = \mathrm{tp}^{\mathfrak{A}}[b]$. For a fixed C^2 formula in normal form (1) a *φ-ray-type* is a 2-type ρ such that $\models \rho \to \bigvee_{h=1}^{n} \chi_i$. If $\mathfrak{A} \models \rho[a, b]$ for distinct elements a, b, then we say that the pair $\langle a, b\rangle$ is a *φ-ray*. We call a φ-ray-type ρ *φ-invertible* if ρ^{-1} is also a φ-ray-type. We call a 2-type τ *φ-silent* if neither τ nor τ^{-1} is a φ-ray-type.

We now construct an apparatus for describing the "local environment" of elements in structures. Let the φ-ray-types be listed in some fixed order (depending on Σ) as $\rho_1, \ldots, \rho_J$. A *φ-star-type* is an $(J+1)$-tuple $\sigma = \langle \pi, v_1, \ldots, v_J\rangle$, where π is a 1-type over Σ and the v_j are non-negative integers such that $v_j \neq 0$ implies $\mathrm{tp}_1(\rho_j) = \pi$ for all j $(1 \leq j \leq J)$. We denote the 1-type π by $\mathrm{tp}(\sigma)$. To motivate this terminology, suppose $\mathfrak{A}$ is a structure interpreting Σ. For any

$a \in A$, we define $\mathrm{st}^{\mathfrak{A}}(a) = \langle \mathrm{tp}^{\mathfrak{A}}[a], v_1, \ldots, v_J \rangle$,where $v_j = |\{b \in A : b \neq a \text{ and } \mathrm{tp}^{\mathfrak{A}}[a,b] = \rho_j\}|$. Evidently, $\mathrm{st}^{\mathfrak{A}}[a]$ is a star-type; we call it the *φ-star-type of a* in $\mathfrak{A}$, and say that *a realizes* $\mathrm{st}^{\mathfrak{A}}[a]$. Intuitively, the star-type of an element records the number of rays of each type emitted by that element. It helps to think, informally, of a star-type σ as *emitting* a collection of rays of various types, and of nodes as *accepting* rays. When φ is known from a context or arbitrary, we will simply write ray-, invertible-, silent- or star-type instead of φ-ray-, φ-invertible-, φ-silent- or φ-star-type. We say that a structure $\mathfrak{A}$ *realizes* a set of 2-types (resp. star-types) Φ if every pair of nodes (resp. every node) in $\mathfrak{A}$ realizes a 2-type (resp. a star-type) from Φ. Importance of the above notions of 2-, ray- and star-types is summarized in the following.

▶ **Proposition 11.** *Let $\mathfrak{A}$ be a structure such that $\mathfrak{A} \models \varphi$. If $\mathfrak{B}$ is a structure interpreting the same signature, and realizing the same set of 2-types and the same set of star-types as $\mathfrak{A}$, then $\mathfrak{B} \models \varphi$.*

Thus, the satisfiability of C^2 formulae is invariant under arbitrary transformations of structures that preserve sets of realized 2-types and star-types. Our transformations are more constrained; for every element of a model they preserve its star-type by only allowing changes of targets of emitted ray-types. Special care must be taken in order not to emit a ray from a source node to a node which already emits a ray back to the source node. Therefore we introduce a restriction allowing to only modify rays that are invertible (rigidity), and another restriction that a node cannot emit an invertible ray-type and another (invertible- or not) ray-type to two nodes with the same 1-type (superchromaticity). This way, we may select an invertible ray-type τ, edges $\tau(e_1, e)$ $\tau(e_1', e')$ and replace them by edges $\tau(e_1', e)$ and $\tau(e_1, e')$ preserving star-types of all involved nodes and not introducing duplicate rays. Furthermore, during the entire procedure we employ additional precautions to preserve both linear order and its successor.

6.2 Towards Vass-completeness of $\mathrm{C}^2[\leq, succ, \sim_k, \pi_{bin}]$

Fix a $\mathrm{C}^2[\leq, succ, \sim_k, \pi_{bin}]$ formula φ in normal form (1) and its interpretation $\mathfrak{A}$. We say that $\mathfrak{A}$ is *φ-rigid* if $\mathfrak{A} \models a \sim_k b$ implies that $\langle a, b \rangle$ is an invertible ray. We say that φ is rigid if all models of φ are φ-rigid. Define ω_k as $\forall x \exists^{\leq k} y. x \sim_k y$. Formulae φ and $\varphi \wedge \omega_k$ are equivalent over $\mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$. Moreover, the latter formula is rigid. We say that $\mathfrak{A}$ is *φ-semichromatic* if no ray is emitted and accepted by nodes of the same 1-type. We say that $\mathfrak{A}$ is *φ-superchromatic* if it is φ-semichromatic and no element emits two or more rays at least one of which is invertible, having the same absorption-type as each other. We say that φ is *φ-semichromatic* (resp. *φ-superchromatic*) if all models of φ are φ-semichromatic (resp. *φ-superchromatic*). The proof of the following lemma is standard (see [10]).

▶ **Lemma 12.** *There is a C^2 formula χ_φ such that φ and $\varphi \wedge \chi_\varphi$ are equisatisfiable on $\mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$ and $\varphi \wedge \chi_\varphi$ is superchromatic. Moreover, if φ is rigid then $\varphi \wedge \chi_\varphi$ is so.*

Now we define formulae that encode $\sim_k$. Fix a set of star-types $\mathfrak{st}$. For $\sigma, \rho \in \mathfrak{st}$ we write $\sigma \sim_k \rho$ if there exists an invertible ray type τ such that $\tau \in \sigma$, $\tau^{-1} \in \rho$ and $\sim_k (x, y) \in \tau$.

Let $\mathfrak{A}$ be a rigid $\mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$-structure over $\mathfrak{st}$. Structure $\mathfrak{A}$ consists of disjoint substructures, each generated by an equivalence class of $\sim_k^{\mathfrak{A}}$. We call such a substructure a *class* in $\mathfrak{A}$. For a class $\mathfrak{C}$ in $\mathfrak{A}$ we call the set $\{\sigma \in \mathfrak{st} \mid \sigma \text{ is realized in } \mathfrak{C}\}$ the *class type* of $\mathfrak{C}$ and denote it by $\mathfrak{ct}(\mathfrak{C})$. Thus $\mathfrak{ct}(\mathfrak{C})$ is a subset of $\mathfrak{st}$. However, not every subset of $\mathfrak{st}$ corresponds to a class type in a $\mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$-structure. A subset $\mathfrak{ct}$ of $\mathfrak{st}$ is called a *class type* wrt. $\mathfrak{st}$ if there is a bijection b from $\mathfrak{c}$ to the k-clique $\mathrm{K}_k = (\mathrm{V}, \mathrm{E})$ such that $(\mathsf{b}(\sigma), \mathsf{b}(\rho)) \in \mathrm{E}$ if and only if $\sigma \sim_k \rho$. Thus, we may identify $\mathfrak{ct}$ with a relational structure, a

clique, being an equivalence class of $\sim_k$. Observe that if $\mathfrak{C}$ is a class in $\mathfrak{A}$ then $\mathfrak{ct}(\mathfrak{C})$ is a class wrt. $\mathfrak{st}$. Thus every class wrt. $\mathfrak{st}$ is potentially a class in some word over $\mathfrak{st}$. Since the size of each class type is bounded by k, the number of class types is bounded by $\binom{|\mathfrak{st}|}{k}$. For any $e \in \mathfrak{C}$ we denote with $\mathfrak{ct}^{\mathfrak{A}}(e)$ its class type in $\mathfrak{A}$, equal to $\mathfrak{ct}(C)$.

Having the above definitions at hand, we may define a two-variable formula $\psi^{\mathfrak{st}}$ that specifies necessary conditions for $\sim_k$ to interpret a bounded equivalence relation in a structure that realizes $\mathfrak{st}$. Formula $\psi^{\mathfrak{st}}$ expresses that every node has precisely one class type, that two nodes connected by $\sim_k$ relation share the same class type, and that a node with a class type $\mathfrak{c}$ realizes some star-type $\sigma \in \mathfrak{c}$. The last property together with φ-semichromaticity implies that star-types of elements within every equivalence class are unique. The entire formula implies that for every node in a structure we may find a set of nodes that together could form an equivalence class. Indeed, we say 'could' since it is not necessary the immediate case, and forming equivalence class may require structure transformations.

For φ in normal form (1) by $\mathfrak{st}(\varphi)$ denote the set of star-types compatible with φ.

▶ **Lemma 13.** *Any model of a* $\mathrm{C}^2[\leq, succ, \sim_k, \pi_{bin}]$ *formula* φ *can be expanded to a model of* $\psi^{\mathfrak{st}(\varphi)}$ *by interpreting fresh unary predicates.*

For a fixed φ to be checked for satisfiability, we set $\varphi_{tr} ::= \varphi \wedge \omega_\varphi \wedge \chi_{\varphi\wedge\omega_\varphi} \wedge \psi^{\mathfrak{st}(\varphi\wedge\omega_\varphi\wedge\chi_{\varphi\wedge\omega_\varphi})}$.

▶ **Lemma 14.** *If a* $\mathrm{C}^2[\leq, succ, \sim_k, \pi_{bin}]$ *formula* φ *is satisfiable in* $\mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$ *then the translation* φ_{tr} *is satisfiable in* $\mathcal{W}(\leq, succ, \pi_{bin})$.

Proof. Let $\mathfrak{A}$ be a model of φ such that $\mathfrak{A} \in \mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$. We will expand $\mathfrak{A}$ by interpreting some fresh unary predicates to obtain a model of φ_{tr}. First, observe that $\mathfrak{A}$ models ω_φ, as each equivalence class of $\sim_k^{\mathfrak{A}}$ has at most k elements. Using Lemma 12, after interpreting some fresh unary predicates, $\mathfrak{A}$ becomes a model of $\chi_{\varphi\wedge\omega_\varphi}$. Then, using Lemma 13, again by interpreting some fresh unary predicates, $\mathfrak{A}$ becomes a model of $\psi^{\mathfrak{st}(\varphi\wedge\omega_\varphi\wedge\chi_{\varphi\wedge\omega_\varphi})}$. The obtained structure remains in class $\mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$ and thus also in $\mathcal{W}(\leq, succ, \pi_{bin})$ and satisfies φ_{tr}. ◀

We now define structure transformations. First, we define a *switch*, whose aim is only to preserve the order of elements. Let us write $a \ll^{\mathfrak{A}} b$ iff $a \leq^{\mathfrak{A}} b$ holds and $succ^{\mathfrak{A}}(a, b)$ does not.

▶ **Definition 15.** *Let* $\mathfrak{A}$ *be a* $\mathcal{W}(\leq, succ, \pi_{bin})$ *structure and* e_1, e, e', *be elements of* A *such that* $e_1 \ll^{\mathfrak{A}} e$ *and* $e \ll^{\mathfrak{A}} e'$. *Define* (e_1, e, e')–switch of $\mathfrak{A}$ *as the structure* $\mathfrak{B}$ *which is identical to* $\mathfrak{A}$ *with the exception that* $\mathrm{tp}^{\mathfrak{B}}(e_1, e) = \mathrm{tp}^{\mathfrak{A}}(e_1, e')$ *and* $\mathrm{tp}^{\mathfrak{B}}(e_1, e') = \mathrm{tp}^{\mathfrak{A}}(e_1, e)$.

Observe that relative order of e_1, e and e' is preserved after the switch and thus both $succ^{\mathfrak{A}}$ and $\leq^{\mathfrak{A}}$ are preserved. The transformation we use is a sequence of two switches, as described by the following lemma.

▶ **Lemma 16** (Switching lemma)**.** *Let* $\mathfrak{A}$ *be a superchromatic* $\mathcal{W}(\leq, succ, \pi_{bin})$ *structure,* e_1', e_1, e, e' *be elements of* A *such that* $e_1 \ll^{\mathfrak{A}} e$, $e_1' \ll^{\mathfrak{A}} e$, $e \ll^{\mathfrak{A}} e'$, *and 2-types* $\mathrm{tp}^{\mathfrak{A}}(e_1', e')$ *and* $\mathrm{tp}^{\mathfrak{A}}(e_1, e)$ *are both the same invertible ray type. The structure* $\mathfrak{B}$ *obtained by the* (e_1, e, e')–*switch of* $\mathfrak{A}$ *followed by the* (e_1', e, e')–*switch belongs to* $\mathcal{W}(\leq, succ, \pi_{bin})$ *and realizes the same set of star- and 2-types as* $\mathfrak{A}$.

Proof. Structure $\mathfrak{A}$ satisfying assumptions of the Lemma is depicted on Fig 1(left). Note that $\mathrm{tp}^{\mathfrak{A}}(e_1', e)$ and $\mathrm{tp}^{\mathfrak{A}}(e_1, e')$ are silent, as otherwise $\mathfrak{A}$ would violate the superchromaticity condition. E.g. $\mathrm{tp}^{\mathfrak{A}}(e_1', e)$ cannot be a ray type, as $\mathrm{tp}^{\mathfrak{A}}(e_1', e')$ is invertible and $\mathrm{tp}^{\mathfrak{A}}(e') = \mathrm{tp}^{\mathfrak{A}}(e)$. The equality of 1-types hold as a conclusion of $\mathrm{tp}^{\mathfrak{A}}(e_1', e) = \mathrm{tp}^{\mathfrak{A}}(e_1, e')$. In a similar way

$\mathrm{tp}^{\mathfrak{A}}(e, e_1')$ cannot be a ray type, thus the $\mathrm{tp}^{\mathfrak{A}}(e_1', e)$ is silent. In a similar way $\mathrm{tp}^{\mathfrak{A}}(e_1, e')$ can be proven silent. After switching we obtain the structure on Fig 1(right), whose star types and 2-types are the same as in $\mathfrak{A}$. ◀

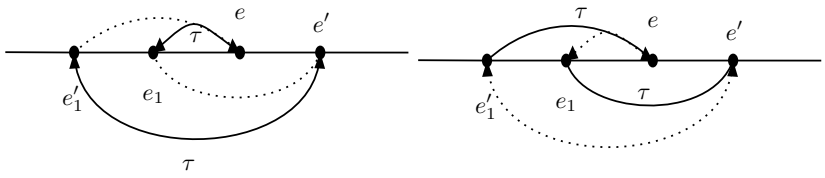

Figure 1 Structure $\mathfrak{A}$ before switching (left) and after switching (right).

The following lemma is the main lemma of this section. There we transform a model of φ_{tr} to another model, where $\sim_k$ is interpreted as a bounded equivalence relation, and where the order is preserved. We decompose the model into substructures generated by elements connected by *succ* and sharing the same class type (thus any class type also decomposes into components). We show that elements within the same component in the model are necessarily connected by $\sim_k$ predicate. Then we employ structure transformations defined above (*i.e.* switches) to show that elements of distinct components of the same class type can be pairwise connected by $\sim_k$ to form equivalence classes.

▶ **Lemma 17.** *If the formula φ_{tr} is satisfiable in $\mathcal{W}(\leq, succ, \pi_{bin})$ then the formula φ is satisfiable in $\mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$.*

Proof. Let $\mathfrak{A}$ be a finite model of φ_{tr} such that $\mathfrak{A} \in \mathcal{W}(\leq, succ, \pi_{bin})$. We will transform $\mathfrak{A}$ to a $\mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$ structure while ensuring that every element of $\mathfrak{A}$ retains its star-type and the set of realized 2-types is preserved. Since $\mathfrak{A} \models \varphi$, by Proposition 11, the obtained structure will still be a model of φ. Observe that φ_{tr} ensures reflexivity and symmetry of $\sim_k$. Thus to obtain a $\mathcal{W}(\leq, succ, \sim_k, \pi_{bin})$ structure we only need to make $\sim_k$ transitive. During the transformation the linear order (that is both $succ^{\mathfrak{A}}$ and $\leq^{\mathfrak{A}}$) remains fixed, while particular 2-types emitted and accepted by structure nodes may change.

Recall that we may identify each class type $\mathfrak{c}$ with a relational structure (a clique). By *component* of $\mathfrak{c}$ we mean any maximal subgraph $\mathfrak{d}$ of $\mathfrak{c}$ such that any node of $\mathfrak{d}$ emits a *succ* edge to some other node of $\mathfrak{d}$ Thus graph $\mathfrak{c}$ consists of (at most k) linearly ordered components, each consisting of at most k elements. Let $\sigma_1, \dots \sigma_l$ be all nodes of $\mathfrak{d}$ listed in order *succ* (all these star-types are distinct as all star-types in any class-type are distinct). Since $\mathfrak{A} \models \psi^{\mathfrak{st}(\varphi \wedge \omega_\varphi \wedge \chi_{\varphi \wedge \omega_\varphi})}$, components of class-types correspond to substructures of $\mathfrak{A}$ in the following way. If $e_1 \in \mathfrak{A}$ is such that $\mathfrak{ct}(e_1) = \mathfrak{c}$ and $\mathrm{st}^{\mathfrak{A}}(e_1) = \sigma_1$ then there exists $l-1$ nodes $e_2, \dots e_l \in \mathfrak{A}$ such that $\mathfrak{ct}(e_i) = \mathfrak{c}$, $\mathrm{st}^{\mathfrak{A}}(e_i) = \sigma_i$ for $i \in \{1, \dots, l\}$, and $succ^{\mathfrak{A}}(e_i, e_{i+1})$ for $i \in \{1, \dots, l-1\}$. By definition of $\mathfrak{d}$ we thus have $e_i \sim_k^{\mathfrak{A}} e_{i+1}$ for $i \in \{1, \dots, l-1\}$. We call the substructure $\mathfrak{D}$ of $\mathfrak{A}$ generated by $e_1, \dots, e_l$ a *component* of $\mathfrak{A}$ corresponding to $\mathfrak{d}$. We define $\mathfrak{co}(\mathfrak{D}) = \mathfrak{d}$ (the *component-type* of $\mathfrak{D}$) and $\mathfrak{ct}(\mathfrak{D}) = \mathfrak{c}$ (the *class-type* of $\mathfrak{D}$).

We will transform $\mathfrak{A}$ so to form equivalence classes of $\sim_k$. These classes will be k-cliques composed of components. Thus, we need to ensure that two conditions hold:

- if two nodes belong to the same component then they are connected by $\sim_k$ edge,
- for every component $\mathfrak{D}_i$ of $\mathfrak{A}$ such that all components of $\mathfrak{ct}(\mathfrak{D}_i)$ listed in order are $\mathfrak{d}_1, \dots \mathfrak{d}_i, \dots \mathfrak{d}_l$, for some numbers i and l, we have the following. There exist l components $\mathfrak{D}_1, \dots \mathfrak{D}_i, \dots \mathfrak{D}_l$ of $\mathfrak{A}$ such that $\mathfrak{co}(\mathfrak{D}_i) = \mathfrak{d}_i$ and if $e_i \in D_i$ and $e_j \in D_j$ then $e_i \sim_k^{\mathfrak{A}} e_j$, for some numbers i, j.

First, we will show that every two elements of a given component of $\mathfrak{A}$ are related by $\sim_k^{\mathfrak{A}}$, *i.e.* that every component of $\mathfrak{A}$ is a clique. We will consider components of $\mathfrak{A}$ in the order defined by $\leq^{\mathfrak{A}}$, assuming that all components visited so far satisfy the required property. Let $\mathfrak{D}$ be a component of $\mathfrak{A}$ currently under inspection, let $\mathfrak{c}$ be the class type of $\mathfrak{D}$ and let $\mathfrak{d}$ be the component type of $\mathfrak{D}$. Take any $a, b \in \mathfrak{D}$ such that $a \leq^{\mathfrak{A}} b$. Let the star-types of a, b in $\mathfrak{A}$ be resp. σ_a and σ_b. Ad absurdum, assume that $a \not\sim_k^{\mathfrak{A}} b$ holds. Since a and b belong to the same component $\mathfrak{D}$, star-types σ_a and σ_b belong to the component $\mathfrak{d}$. Since $\mathfrak{d}$ is a clique graph, there exists ray-type τ such that $x \leq y \in \tau$, $x \sim_k y \in \tau$, $\tau \in \sigma_a$, and $\tau^{-1} \in \sigma_b$. Since the star-type of b is σ_b, there exists $a' \in \mathfrak{A}'$ such that $\mathrm{tp}^{\mathfrak{A}}(a', b) = \tau$. Since $\mathfrak{A} \models \psi^{\mathfrak{st}(\varphi \wedge \omega_\varphi \wedge \chi_{\varphi \wedge \omega_\varphi})}$, class-type of a' is the same as the class-type of b, *i.e.* $\mathfrak{ct}(a') = \mathfrak{c}$. Since 1-types within class-types are unique, we have $\mathrm{st}^{\mathfrak{A}}(a') = \sigma_a$ and a' belongs to a component $\mathfrak{D}'$ of $\mathfrak{A}$ such that the component type of $\mathfrak{D}'$ is $\mathfrak{d}$, $\mathfrak{D}' \neq \mathfrak{D}$ and $\mathfrak{D}'$ occurs in $\mathfrak{A}$ earlier (wrt. $\leq^{\mathfrak{A}}$) than $\mathfrak{D}$. By the inductive assumption all elements of $\mathfrak{D}'$ are connected by $\sim_k$. Since the component type of $\mathfrak{D}'$ is $\mathfrak{d}$, there exits a $b' \in \mathfrak{D}'$ such that $\mathrm{st}^{\mathfrak{A}}(b') = \sigma_b$. Thus $\mathrm{tp}^{\mathfrak{A}}(a', b') = \tau$. But, simultaneously $\mathrm{tp}^{\mathfrak{A}}(a', b) = \tau$. Because of superchromaticity this can only be true if $b' = b$, but these nodes belong to disjoint substructures $\mathfrak{D}'$ and $\mathfrak{D}$ of $\mathfrak{A}$. Contradiction. Thus $a \sim_k^{\mathfrak{A}} b$ holds implying, that any two elements of the same component of $\mathfrak{A}$ are related by $\sim_k^{\mathfrak{A}}$.

Now we must switch edges of $\mathfrak{A}$ so to ensure that elements of distinct components are connected by $\sim_k$ edges to form equivalence classes of $\sim_k^{\mathfrak{A}}$. We traverse components of $\mathfrak{A}$ in the order defined by $succ^{\mathfrak{A}}$ restoring $\sim_k$ relations between their nodes, when necessary, by employing Lemma 16. ◀

Since the finite satisfiability for $\mathrm{C}^2[\leq, succ, \pi_{bin}]$ is Vass-complete [12], by Lemma 14 and Lemma 17 we immediately conclude:

▶ **Theorem 18.** $\mathrm{C}^2[\leq, succ, \sim_k, \pi_{bin}]$ *is* Vass*-complete.*

7 Conclusions

We considered counting extensions of the two-variable logic on finite data-words. While our main logic, namely $\mathrm{C}^2[\leq, succ, \sim, \pi_{bin}]$ turned out to be undecidable, we identified several decidable sub-logics, with complexities ranging from NExpTime to Vass, depending on the allowed binary relations in the vocabularies. We hope that the outcome of the paper might be interesting for the two-variable community and that the established decidability results can be later generalised to capture even more expressive forms of quantitative properties.

References

1 Rajeev Alur, Pavol Cerný, and Scott Weinstein. Algorithmic analysis of array-accessing programs. *ACM Trans. Comput. Log. 2012*, 2012. doi:10.1145/2287718.2287727.

2 Henrik Björklund and Mikolaj Bojańczyk. Shuffle Expressions and Words with Nested Data. In *MFCS 2007*, 2007. A version with an appendix available at https://www.mimuw.edu.pl/~bojan/upload/confmfcsBjorklundB07.pdf. doi:10.1007/978-3-540-74456-6_66.

3 Henrik Björklund and Thomas Schwentick. *Class-Memory Automata Revisited*, pages 201–215. Springer, 2017. doi:10.1007/978-3-319-48317-7_12.

4 Mikolaj Bojańczyk, Claire David, Anca Muscholl, Thomas Schwentick, and Luc Segoufin. Two-variable logic on data words. *ACM Trans. Comput. Log. 2011*, 2011. doi:10.1145/1970398.1970403.

5 Mikolaj Bojańczyk and Slawomir Lasota. An extension of data automata that captures XPath. *LMCS 2012*, 2012. doi:10.2168/LMCS-8(1:5)2012.

6 Benedikt Bollig. An Automaton over Data Words That Captures EMSO Logic. In *CONCUR 2011*, 2011. doi:10.1007/978-3-642-23217-6_12.

7 Benedikt Bollig, Peter Habermehl, Martin Leucker, and Benjamin Monmege. A robust class of data languages and an application to learning. *LMCS 2014*, 2014. doi:10.2168/LMCS-10(4:19)2014.

8 Patricia Bouyer. A logical characterization of data languages. *Inf. Process. Lett. 2002*, 2002. doi:10.1016/S0020-0190(02)00229-6.

9 Patricia Bouyer, Antoine Petit, and Denis Thérien. An algebraic approach to data languages and timed languages. *Inf. Comput. 2003*, 2003. doi:10.1016/S0890-5401(03)00038-5.

10 Witold Charatonik, Yegor Guskov, Ian Pratt-Hartmann, and Piotr Witkowski. Two-variable First-Order Logic with Counting in Forests. In *LPAR 2018*, 2018.

11 Witold Charatonik, Ian Pratt-Hartmann, and Piotr Witkowski. Two-Variable Logic with Counting and Data-Trees. Submitted. Available at http://www.cs.man.ac.uk/~ipratt/papers/logic/c21d1e.pdf.

12 Witold Charatonik and Piotr Witkowski. Two-variable Logic with Counting and a Linear Order. *LMCS 2016*, 2016. doi:10.2168/LMCS-12(2:8)2016.

13 Taolue Chen, Fu Song, and Zhilin Wu. Formal Reasoning on Infinite Data Values: An Ongoing Quest. In *SETSS 2016, Chongqing, China, March 28 - April 2, 2016*, 2016. doi:10.1007/978-3-319-56841-6_6.

14 Conrad Cotton-Barratt. *Using Class Memory Automata in Algorithmic Game Semantics*. PhD thesis, University of Oxford, UK, 2016.

15 Conrad Cotton-Barratt, Andrzej S. Murawski, and C.-H. Luke Ong. Weak and Nested Class Memory Automata. In *LATA 2015*, 2015. doi:10.1007/978-3-319-15579-1_14.

16 Wojciech Czerwinski, Slawomir Lasota, Ranko Lazic, Jérôme Leroux, and Filip Mazowiecki. The reachability problem for Petri nets is not elementary. In *STOC 2019*, 2019. doi:10.1145/3313276.3316369.

17 Loris D'Antoni. In the maze of data languages. *CoRR*, 2012. URL: http://arxiv.org/abs/1208.5980.

18 Stéphane Demri, Diego Figueira, and M. Praveen. Reasoning about Data Repetitions with Counter Systems. *LMCS 2016*, 2016. doi:10.2168/LMCS-12(3:1)2016.

19 Stéphane Demri and Ranko Lazic. LTL with the freeze quantifier and register automata. *ACM Trans. Comput. Log. 2009*, 2009. doi:10.1145/1507244.1507246.

20 Kousha Etessami, Moshe Y. Vardi, and Thomas Wilke. First-Order Logic with Two Variables and Unary Temporal Logic. *Inf. Comput. 2002*, 2002. doi:10.1006/inco.2001.2953.

21 Diego Figueira. A Decidable Two-Way Logic on Data Words. In *LICS 2011*, 2011. doi:10.1109/LICS.2011.18.

22 Daniel Genkin, Michael Kaminski, and Liat Peterfreund. A note on the emptiness problem for alternating finite-memory automata. *Theor. Comput. Sci. 2014*, 2014. doi:10.1016/j.tcs.2014.01.020.

23 Erich Grädel and Martin Otto. On Logics with Two Variables. *Theor. Comput. Sci. 1999*, 1999. doi:10.1016/S0304-3975(98)00308-9.

24 Ahmet Kara. *Logics on data words: Expressivity, satisfiability, model checking*. PhD thesis, Technical University of Dortmund, Germany, 2016. URL: http://hdl.handle.net/2003/35216.

25 Jérôme Leroux and Sylvain Schmitz. Reachability in Vector Addition Systems is Primitive-Recursive in Fixed Dimension. In *LICS 2019*, 2019. doi:10.1109/LICS.2019.8785796.

26 Marvin L. Minsky. *Computation: Finite and Infinite Machines*. Prentice Hall, 1967.

27 Angelo Montanari, Marco Pazzaglia, and Pietro Sala. Metric propositional neighborhood logic with an equivalence relation. *Acta Inf. 2016*, 2016. doi:10.1007/s00236-016-0256-3.

28 Matthias Niewerth. *Data definition languages for XML repository management systems*. PhD thesis, Technical University of Dortmund, Germany, 2016.

29 Ian Pratt-Hartmann. The Two-Variable Fragment with Counting Revisited. In *WoLLIC 2010*, 2010. doi:10.1007/978-3-642-13824-9_4.

30 Ian Pratt-Hartmann. The two-variable fragment with counting and equivalence. *Math. Log. Q. 2015*, 2015. doi:10.1002/malq.201400102.

31 Sebastian Rudolph. Presburger Concept Cardinality Constraints in Very Expressive Description Logics - Allegro sexagenarioso ma non ritardando. In *Description Logic, Theory Combination, and All That - Essays Dedicated to Franz Baader on the Occasion of His 60th Birthday*, 2019. doi:10.1007/978-3-030-22102-7_25.

32 Fu Song and Zhilin Wu. On temporal logics with data variable quantifications: Decidability and complexity. *Inf. Comput. 2016*, 2016. doi:10.1016/j.ic.2016.08.002.