Successor-Invariant First-Order Logic on Classes of Bounded Degree (Extended Abstract)

Julien Grange
Université de Rennes 1
julien.grange@inria.fr

Abstract

We study the expressive power of successor-invariant first-order logic, which is an extension of first-order logic where the usage of a successor relation on the vertices of the graph is allowed, as long as the validity of formulas is independent on the choice of a particular successor.

We show that when the degree is bounded, successor-invariant first-order logic is no more expressive than first-order logic.

1 Introduction

1.1 The Notion of Invariance

The notion of order-invariance is a very natural one. For instance, when writing down a set by extension (*i.e.* as a list of all its elements), one has to choose a particular order in which those elements are to appear. The sole purpose of this order, however, is to comply to the linear constraint of writing on a piece of paper; one can write the same elements in a different order without affecting the nature of the represented set. In this sense, one sees that the definition by extension of a set is independent of the order.

To see how this remark applies to computation theory, let us consider a Turing machine $\mathcal{M}$ deciding a graph problem. Of course, $\mathcal{M}$ doesn't directly work on abstract graphs, but rather on their representations, *e.g.* their adjacency matrices. A single graph can have many distinct representations, however $\mathcal{M}$ decides a graph problem; hence it must either accept all of these representations, or reject all of them.

The translation of this idea to logic yields the notion of *invariant logics*. Starting from some logic (let's say first-order logic, FO) on the vocabulary of graphs, consisting of a single binary predicate, we allow sentences to use (through the addition of another binary predicate $<$) a linear order on the vertices of the graphs, provided that given a finite graph, such a sentence is either always true no matter what order we are given on its vertices, or always false. This indeed corresponds to the previous idea that, while we would like our sentences to make the most out of the way a graph is represented, *i.e.* not only as an abstract structure (as it is usually though of in logic) but as one of its adjacency matrices, we still want to express properties of graphs, which must be independent of any particular representation. Order-invariant FO, $<$-inv FO, is precisely the set of sentences that use the additional order in an invariant fashion.

One might want to restrict the power of the additional relation. A way to do this is not to grant a complete access to a linear order on the vertices, but only to the successor relation associated to such an order. This yields Succ-inv FO, successor-invariant first-order logic.

1.2 Motivations

To illustrate the relevance of these invariant logics in computer science, let's see how they appear in database theory and in descriptive complexity.

In database theory, one studies the set of queries which are expressible in a given language (*i.e.* a logic, if we look at databases as relational structures). Given that a real-world database is stored on disk, one can infer an order on the elements of the database from the specifics of the disk. If one is to use this order in their queries, one should be cautious: this order is orthogonal to the semantics, thus it would be undesirable if the queries were to depend upon it. This would in particular entail that upon a copy of the database (which has no reason to preserve the order on disk), the same query may yield different results on two instances of the same database. The queries that are harmless in this regard are precisely the order- or successor-invariant ones.

Descriptive complexity aims at establishing correspondences between complexity classes, defined as classes of problems that can be solved with some limitations on the space/time resources available to a Turing machine, and logics in which these problems can be defined. It turns out invariance is central in this matter, as one may already suspect when looking at our introductory example, where invariance captures the notion of independence wrt. the representation of objects on a Turing machine's tape. For instance, the Immerman-Vardi Theorem [Immerman, 1986] states that the problems in PTIME are exactly those which can be described in Succ-inv LFP, where LFP is defined from FO by the addition of a least-fixed-point operator. Note that this correspondence doesn't yield a logic for PTIME, as the successor- and order-invariant expansions of LFP (and FO) don't have a recursive syntax. In this context, sharpening our intuition of invariant expansions (and finding recursive formalisms having

the same expressive power) could have direct implications on our understanding of classical complexity classes.

1.3 The Expressive Power of Invariance

Now that we have seen several motivations to these definitions, let us take a look at the expressive power of our two formalisms, $<$-inv FO and Succ-inv FO.

Given that the order (resp. successor) at hand must be used only in an invariant way, it is not obvious whether these expansions grant any additional expressive power to plain FO. Perhaps surprisingly, it turns out that some properties beyond the scope of FO are now definable; see Section 5.2 of [Libkin, 2004] for an example due to Gurevich separating FO from $<$-inv FO, and [Rossman, 2007] for the strengthened result that on finite structures, Succ-inv FO is strictly more expressive than FO.

It is interesting to note that the two examples above, as well as the few other known separating examples, are based on dense classes of structures. Two natural questions arise at this point:

1. does the addition of an invariant order or successor still bring expressive power to FO on sparse classes of graphs (*e.g.* on classes of bounded degree, bounded treewidth, etc.)?
2. on a given class of graphs, can we find a logic that has the same expressive power as $<$ -inv FO and Succ-inv FO?

As we have discussed previously, these questions have important counterparts both in complexity and database theory.

Some progress has been made toward answering these questions. In particular, it has been shown that $<$-inv FO (hence Succ-inv FO too) collapses to FO on trees [Benedikt and Segoufin, 2009] and on hollow trees [Grange and Segoufin, 2020]. On classes of bounded degree and in the setting of decomposable structures (which includes the case of classes of bounded treewidth), it is known that Succ-inv FO and $<$-inv FO are included (in the sense of their expressive power) in MSO [Benedikt and Segoufin, 2009; Elberfeld *et al.*, 2016].

In the present paper, we take a step towards these goals by proving that when the degree is bounded, Succ-inv FO has the same expressive power as FO.

2 The Collapse and (an Overview of) its Proof

Let us now dive into the specifics of our result. We state the main theorem in Section 2.1 after introducing formally Succ-inv FO. In the subsequent sections, we give an overview of the successive steps of the proof.

In this extended abstract, we will only consider finite graphs (seen as structures over the binary predicate E); the results presented here can obviously be extended without difficulty to the general case of finite relational structures.

2.1 Definition of Succ-inv FO

A binary relation on a finite set X is a **successor relation on** X if it is the graph of a circular permutation of X, *i.e.* a bijective function from X to X with a single orbit. This differs from the standard notion of successor in that there is neither minimal nor maximal element; this is convenient, as these are two irregularities with which we won't have to deal. Note that this choice doesn't have any impact on the expressive power of Succ-inv FO, as both notions of successors are interdefinable in FO.

We use the standard definition of first-order logic $\mathrm{FO}(\Sigma)$, the vocabulary Σ being either $\{E\}$ or $\{E, S\}$, where the binary predicate E is the edge predicate, and the binary predicate S is our successor predicate. Following the tradition in logic, we denote our structures (*i.e.* our graphs) by calligraphic upper-case letters, while their universes (*i.e.* their vertex set) are denoted by the corresponding standard upper-case letters; for instance, G is the vertex set of the graph $\mathcal{G}$. A graph $\mathcal{G}$ with a successor relation S on its vertices, seen as an $\{E, S\}$-structure, will be denoted as $(\mathcal{G}, S)$. For convenience, we sometimes identify a predicate and its interpretation in a structure.

Definition 1 (Succ-inv FO)**.** *A sentence $\varphi \in \mathrm{FO}(\{E, S\})$ is said to be* ***successor-invariant*** *if for every finite graph $\mathcal{G}$, and every successor relations S_1 and S_2 on G, $(\mathcal{G}, S_1) \models \varphi$ iff $(\mathcal{G}, S_2) \models \varphi$. We can then omit the interpretation for the predicate S, and if $(\mathcal{G}, S) \models \varphi$ for any (every) successor S, we write $\mathcal{G} \models \varphi$.*

The set of successor-invariant sentences is denoted Succ-inv FO.

In what follows, we only consider finite graphs: if we require a sentence to be successor-invariant on every finite and infinite graph, the Interpolation theorem ensures that this sentence is equivalent to a sentence which doesn't use the successor relation, and the notion of successor-invariance becomes meaningless.

For a class $\mathcal{C}$ of finite graphs, we say that

$$\text{Succ-inv FO} = \text{FO on } \mathcal{C}$$

if the properties of $\mathcal{C}$ definable in Succ-inv FO and in FO are the same. In other words, if for every $\varphi \in$ Succ-inv FO, there exists some $\bar{\varphi} \in$ FO such that

$$\forall \mathcal{G} \in \mathcal{C}, \quad \mathcal{G} \models \varphi \quad \text{iff} \quad \mathcal{G} \models \bar{\varphi}\,.$$

The reverse inclusion, *i.e.* FO $\subseteq$ Succ-inv FO, always holds and needs no verification.

We are now ready to state our main result:

Theorem 1. *For every class $\mathcal{C}$ of finite graphs of bounded degree,*

$$\text{Succ-inv FO} = \text{FO } \textit{on } \mathcal{C}\,.$$

The remainder of this article is dedicated to an overview of the proof of Theorem 1.

2.2 Overarching Strategy

Let's consider a class $\mathcal{C}$ of finite graphs of degree at most d.

In order to prove that Succ-inv FO $\subseteq$ FO on $\mathcal{C}$, we adopt a standard strategy. Let's consider a property of $\mathcal{C}$ definable by a formula φ in Succ-inv FO. To prove Theorem 1, we need to show that FO is able to make a distinction between graphs that satisfy φ and graphs that don't. It is thus natural to rely on the well-know notion of FO-similarity, which measures the point to which two graphs look alike in the eyes of FO.

Definition 2 (FO-similarity)**.** *Given two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, we write $\mathcal{G}_1 \equiv_k^{\mathrm{FO}} \mathcal{G}_2$, and say that that $\mathcal{G}_1$ and $\mathcal{G}_2$ are* ***FO-similar at depth*** *k, if $\mathcal{G}_1$ and $\mathcal{G}_2$ satisfy the same* FO*-sentences of quantifier rank at most k. We adopt the same vocabulary and notation (namely, $(\mathcal{G}_1, S_1) \equiv_k^{\mathrm{FO}} (\mathcal{G}_2, S_2)$) for graphs with successors.*

We can now rephrase our goal. Let k be the quantifier rank of φ. We aim to show the existence of a large enough $n \in \mathbb{N}$ (which depends on k) such that, given two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ of $\mathcal{C}$ that are FO-similar at depth n, we are able to construct a successor relation S_1 on $\mathcal{G}_1$ and S_2 on $\mathcal{G}_2$ such that $(\mathcal{G}_1, S_1)$ and $(\mathcal{G}_2, S_2)$ are FO-similar at depth k; hence, agree on φ.

In other words, the property defined by φ is also a union of equivalence classes for $\equiv_n^{\mathrm{FO}}$. Finite model theory ensures that $\equiv_n^{\mathrm{FO}}$ has finite index and that each one of its classes is FO-definable; thus the union of FO-sentences defining these classes in an FO-sentence defining the same property as φ. This concludes the proof that Succ-inv FO $\subseteq$ FO on $\mathcal{C}$.

What remains is to show how to choose n based on k, and how to construct two successors S_1 and S_2 which maintain FO-similarity between $\mathcal{G}_1$ and $\mathcal{G}_2$.

2.3 Locality

Remember that once we've constructed S_1 and S_2, it will remain to show that $(\mathcal{G}_1, S_1)$ and $(\mathcal{G}_2, S_2)$ are FO-similar at depth k. Proving the similarity of two structures can easily get out of hand; to ease the reasoning, we rely on the local behavior of FO.

When the degree is bounded, the expressive power of FO can be characterized in terms of neighborhoods; basically, FO can exactly express that in a graph, there is a certain number of occurrences (up to some threshold) of some subgraphs of small diameter.

To make this characterization precise, we need to introduce the notion of neighborhood.

Definition 3 (Neighborhood types)**.** *Let $\mathcal{A}$ be either a graph $\mathcal{G}$ or a graph with a successor relation $(\mathcal{G}, S)$, seen as a Σ-structure for the appropriate vocabulary Σ.*

For $r \in \mathbb{N}$ and $a \in A$, the ***r-neighborhood*** *$\mathcal{N}_{\mathcal{A}}^r(a)$ of a is the Σ-substructure of $\mathcal{A}$ induced by the set of vertices at (unoriented) distance at most r from a in $\mathcal{A}$. In the case of a graph with successor, the distance takes into account both E and S.*

The ***r-neighborhood type*** *$\tau = \mathrm{tp}_{\mathcal{A}}^r(a)$ of a is the isomorphism class of its r-neighborhood. We say that τ is a neighborhood type over Σ, and that a is an* ***occurrence*** *of τ. We denote by $|\mathcal{A}|_\tau$ the number of occurrences of τ in $\mathcal{A}$, and we write $[\![\mathcal{A}]\!]_r =^t [\![\mathcal{B}]\!]_r$ to mean that for every r-neighborhood type τ, $|\mathcal{A}|_\tau$ and $|\mathcal{B}|_\tau$ are either equal, or both larger than t.*

With these notions made precise, we can state the Hanf threshold theorem: given the bound d on the degree and the integer k, there exist two integers r and t such that

$$[\![(\mathcal{G}_1, S_1)]\!]_r =^t [\![(\mathcal{G}_2, S_2)]\!]_r \tag{1}$$

is enough for our needs; in other words, (1) entails

$$(\mathcal{G}_1, S_1) \equiv_k^{\mathrm{FO}} (\mathcal{G}_2, S_2)\,.$$

Conversely, for any integer t', we can set n so that our hypothesis $\mathcal{G}_1 \equiv_n^{\mathrm{FO}} \mathcal{G}_2$ entails $[\![\mathcal{G}_1]\!]_r =^{t'} [\![\mathcal{G}_2]\!]_r$.

We have reformulated our problem in the following way: starting from $\mathcal{G}_1$ and $\mathcal{G}_2$ that have the same number of occurrences (up to a threshold t') of every r-neighborhood type, can we construct two successor relations S_1 and S_2 such that $(\mathcal{G}_1, S_1)$ and $(\mathcal{G}_2, S_2)$ still have the same number of occurrences of each r-neighborhood type (which now take the successor relation into account), albeit up to the lesser threshold t?

The construction of such successor relations is the object of the following section.

2.4 Fractal Types

We are now in a position to briefly describe how to construct suitable successor relations S_1 and S_2.

First, we separate the neighborhood types occurring in $\mathcal{G}_1$ and $\mathcal{G}_2$ into two categories:

- on the one hand, the rare neighborhood types, which have few occurrences in $\mathcal{G}_1$ and $\mathcal{G}_2$ (and thus, that have the same number of occurrences in both structures, since $[\![\mathcal{G}_1]\!]_r =^{t'} [\![\mathcal{G}_2]\!]_r$)
- on the other hand, the frequent neighborhood types, which have many occurrences both in $\mathcal{G}_1$ and $\mathcal{G}_2$.

The idea is to construct S_1 and S_2 in the most regular way possible. In an ideal world, one would be able to construct those successor relations such that $\mathrm{tp}^r_{(\mathcal{G}_\epsilon, S_\epsilon)}(a)$, for $\epsilon \in \{1, 2\}$, is completely determined by $\mathrm{tp}^r_{\mathcal{G}_\epsilon}(a)$. This would ensure, as desired, that $[\![(\mathcal{G}_1, S_1)]\!]_r =^t [\![(\mathcal{G}_2, S_2)]\!]_r$.

Of course, there is no hope to find such regular S_ϵ in general, but the heart of the construction is to make sure that most neighborhoods follow this behavior.

Let's consider a frequent neighborhood type τ. By definition, it has enough occurrences in $\mathcal{G}_1$ and $\mathcal{G}_2$ so that we can make sure that (almost everywhere) the S_ϵ-successor and -predecessor of an element of type τ in $\mathcal{G}_\epsilon$ also have type τ. On top of that, we have enough occurrences of τ to make sure that S_ϵ only links vertices that are far enough from one another in the graph.

This very regular neighborhood type is called the **fractal type** of τ.

This is depicted in Figure 1. Note that we require that there is no intersection between neighborhoods of $\mathcal{G}_\epsilon$ appearing in the same fractal neighborhood in $(\mathcal{G}_\epsilon, S_\epsilon)$; this is what gives to fractal neighborhoods their "layered" aspect.

We now know how to deal with most frequent neighborhoods in a regular way. However, there still remain two kind of singularities for us to deal with.

First, we have to deal with occurrences of rare neighborhood types. By definition, there are only a small number of such occurrences. Hence we can "hide" them among occurrences of a given frequent neighborhood type, of which there are many more. This will allow us to give the same treatment to occurrences of rare neighborhood types in $\mathcal{G}_1$ and in $\mathcal{G}_2$, so that their extended type in $(\mathcal{G}_1, S_1)$ and $(\mathcal{G}_2, S_2)$ are the same.

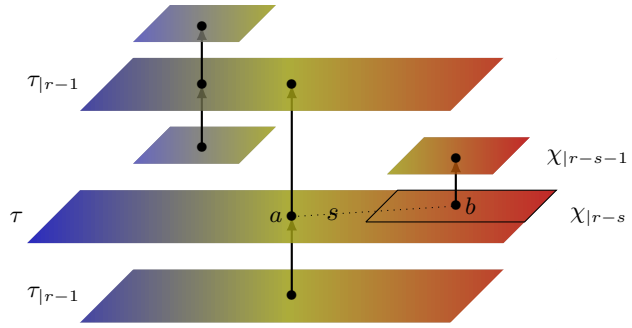

Figure 1: Partial representation of the fractal type of the frequent r-neighborhood τ around a vertex a. The vertical arrows represent S_ϵ, while horizontal planes are neighborhoods in $\mathcal{G}_\epsilon$.

Then, we also have to take care of the junctions between the frequent types: if there exist at least two frequent types, then at some point, an occurrence of some frequent type will have for successor an occurrence of another frequent type. If handled with a bit of care (in particular, by always respecting the layering), this does not cause any issue, as we can make sure that those junctions are treated similarly in $\mathcal{G}_1$ as in $\mathcal{G}_2$.

We illustrate this process by considering a situation where there are three frequent neighborhood types τ_0, τ_1 and τ_2, and one rare neighborhood type χ with two occurrences in $\mathcal{G}_1$ and $\mathcal{G}_2$. At the end of the construction, S_ϵ will (mostly) look like in Figure 2, where the relations of $\mathcal{G}_\epsilon$ have been omitted and the arrows represent S_ϵ, which is indeed a circular successor.

Note that all the elements of neighborhood type τ_1 form a segment wrt. S_ϵ, as well as all the elements of neighborhood type τ_2. The first frequent neighborhood type, τ_0, has a special role in that it is used to embed all the elements of rare neighborhood type (here, χ). Furthermore, and this is not apparent in the figure, two successive elements for S_ϵ are always distant in $\mathcal{G}_\epsilon$.

In the end, we have constructed $\mathcal{S}_1$ and $\mathcal{S}_2$ in such a way that r-neighborhood types created in $(\mathcal{G}_1, S_1)$ and $(\mathcal{G}_2, S_2)$ are the same, and occur the same number of times (up to the threshold t).

As we have seen in Sections 2.2 and 2.3, this is enough to ensure that $\mathcal{G}_1$ and $\mathcal{G}_2$ agree on φ, and in turn that Succ-inv FO $\subseteq$ FO on $\mathcal{C}$, thus concluding the proof of Theorem 1.

3 Conclusion

We have shown that, when the degree is bounded, Succ-inv FO has the same expressive power as plain FO.

There are two main directions in which one could look to extend the present result. One possibility would be to keep looking at classes of bounded degree while climbing up in the ladder of expressive power, and ask whether $<$ -inv FO collapses to FO as well on these classes of graphs. New techniques would be needed, as contrary to what was the case

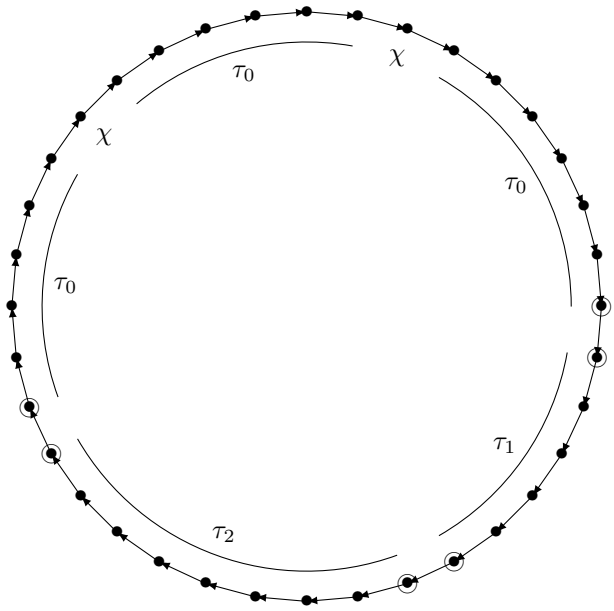

Figure 2: Illustration of S_ϵ when there are three frequent neighborhood types (τ_0, τ_1, τ_3) and one rare neighborhood type (χ) in $\mathcal{G}_\epsilon$. The elements of rare neighborhood type are surrounded by occurrences of the first frequent neighborhood type, τ_0. Junction elements are circled.

with a successor relation, the addition of an order doesn't preserve the bounded degree property. Furthermore, even if $<$-inv FO $=$ FO in this setting, it is not clear whether such orders can be directly constructed. It may be necessary to construct, as in [Benedikt and Segoufin, 2009], a chain of intermediate graphs and orders.

Alternatively, we could change the setting, and study the expressive power of Succ-inv FO on other sparse classes of graphs, *e.g.* on classes of bounded treewidth. Note that the task of constructing appropriate successor relations without any bound on the degree seems much harder; this property was what guaranteed that we could find elements of a given frequent neighborhood type far from each other.

We leave these questions for further research.

References

[Benedikt and Segoufin, 2009] Michael Benedikt and Luc Segoufin. Towards a characterization of order-invariant queries over tame graphs. *J. Symb. Log.*, 2009.

[Elberfeld *et al.*, 2016] Michael Elberfeld, Marlin Frickenschmidt, and Martin Grohe. Order invariance on decomposable structures. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS*, 2016.

[Grange and Segoufin, 2020] Julien Grange and Luc Segoufin. Order-Invariant First-Order Logic over Hollow Trees. In *28th EACSL Annual Conference on Computer Science Logic (CSL 2020)*. Schloss Dagstuhl, 2020.

[Immerman, 1986] Neil Immerman. Relational queries computable in polynomial time. *Inf. Control.*, 1986.

[Libkin, 2004] Leonid Libkin. *Elements of Finite Model Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2004.

[Rossman, 2007] Benjamin Rossman. Successor-invariant first-order logic on finite structures. *J. Symb. Log.*, 2007.