

A NORMAL FORM FOR FIRST-ORDER LOGIC OVER DOUBLY-LINKED DATA STRUCTURES

STEVEN LINDELL

*Department of Computer Science, Haverford College
Haverford, PA, 19041, USA.*

Received 1 January 2006

Accepted 11 June 2007

Communicated by P. Kolaitis

We use singular vocabularies to analyze first-order definability over doubly-linked data structures. Singular vocabularies contain only monadic predicate and monadic function symbols. A class of mathematical structures in any vocabulary can be elementarily interpreted in a singular vocabulary, while preserving notions of total size and degree. Doubly-linked data structures are a special case of bounded-degree finite structures in which there are reciprocal connections between elements, corresponding closely with physically feasible models of information storage. They can be associated with logical models involving unary relations and bijective functions in what we call an invertible singular vocabulary. Over classes of these models, there is a normal form for first-order logic which eliminates all quantification of dependent variables. The paper provides a syntactically based proof using counting quantifiers. It also makes precise the notion of implicit calculability for arbitrary arity first-order formulas. Linear-time evaluation of first-order logic over doubly-linked data structures becomes a direct corollary. Included is a discussion of why these special data structures are appropriate for physically realizable models of information.

Keywords: Doubly-linked data structures; Logical normal forms; physically realizable models; linear time model checking.

1. Introduction

This research grew out of a question involving signatures of first-order structures:

- *What is the simplest vocabulary retaining the full power of first-order logic?*

There are many ways to define simplicity, but we chose “smallest arity” as our yardstick, and wondered if the explicit use of pairing in a structure could somehow be avoided. Part of the motivation came from practical applications, because the basic data structures used in Computer Science rely entirely on nodes which store data and pointers to other nodes. These structures arose as models of data storage subject to the practicalities of physical memory devices. This presents a new question regarding the scalability of data structures:

- *What are the physically realizable asymptotic classes of data structures?*

Though we do not extensively argue a convincing answer for this question, we do take the position that doubly-linked data structures are required. Part of the reason for this is the inherent reversibility of connections in space, which in real memory must comprise actual physical links.

Although using pointers to efficiently represent data has clearly proven its practical value in nearly all areas of computing, describing finite models in a theoretical fashion by unary functional vocabularies seems to have been long neglected until the work of Grandjean [4, 5]. Indeed, this clever idea has proven its utility by a line of research which has culminated in robustly capturing deterministic linear time computation on random access machines by defining unary functions via algebraic recursion schemes [6]. In this paper, we are more interested in how the logical properties of reversibly-linked structures relate to linear-time.

It is well known that structures over any fixed vocabulary can be elementarily interpreted in the language of graphs – i.e. only a single binary relation (with equality) is required.^a But it is actually more transparent to target the language of data structures – a singular vocabulary with monadic predicates and monadic functions (and equality). As we shall see, this interpretation can be naturally constructed in a fashion that conserves total size (the number of tuples) and total degree (how much they cluster).

Distinct from monadic logic, which excludes function symbols, first-order singular logic is expressive enough to interpret elementary definability over any vocabulary whatsoever. The construction is straightforward yet reveals an important fact – in general there will be an unbounded number of references to an unbounded number of nodes. While this presents no problem for purely mathematical studies, it is not realistic for physically scalable models of information. Structures of unbounded-degree cannot be physically realizable. So we restrict our attention to models of information which have a fixed number of symmetric connections between their datum – otherwise known as doubly-linked data structures. While this does not necessarily ensure physical realizability, it comes much closer. The natural vocabulary for these classes is an invertible singular vocabulary in which each function has an inverse. Within this realm, first-order definability is much simpler than it appears. Our main result states that it is possible to rewrite all first-order formulas in a normal form that eliminates all quantified dependencies between variables.

Result: Any first-order formula $\zeta(x_1, \dots, x_k)$ in an invertible singular vocabulary is equivalent to a Boolean combination of:

- (i) atomic formulas: $\alpha(x_1, \dots, x_k)$ &
- (ii) numerical sentences: $\exists^n x \beta(x)$ where β is quantifier-free.

Over finite structures, the numerically quantified sentences can be computed in linear-time, and the quantifier-free formulas in constant-time. With the appropriate

^aA fairly complete treatment of this result appears in my Ph.D. dissertation: “The Logical Complexity of Queries on Unordered Graphs,” University of California at Los Angeles, 1987.

notion of evaluation for formulas with free variables, we obtain a corollary which states that all such first-order formulas are computable in linear-time.

Ideas behind a semantic collapse for first-order queries on bounded-degree structures were motivated from notions of locality due to Hanf [7], as treated in chapter 4 of [9] and chapter 6 of [8]. Ideas for a syntactic collapse of first-order formulas were inspired by Gaifman [3], whose normal form is remarkable for its generality. It applies to all relational structures, but quantified dependencies between variables remain [9, p.60]. Further motivation was obtained from the pioneering result of Seese [13], which demonstrated a linear-time algorithm for first-order sentences over bounded-degree graphs [8, p.103] and [9, p.101]. A related result was presented in [10] in a different logical setting, applying Hanf’s Lemma to first-order sentences, but it did not apply to formulas or discuss linear-time evaluation. Very similar results to ours were attained completely independently in [2].

The outline of the paper is as follows. In the first section we define singulary logic, its syntax and semantics. In the second section we illustrate how any relational structure can be interpreted as a singulary structure via a simple first-order transformation that maintains the natural notion of its total size. In the third section we show that this transformation also maintains a natural notion of total degree. In the fourth section we examine finite domains, and see how doubly-linked data structures are naturally representable as singulary models. The fifth section contains our main result, a normal form for first-order formulas in invertible singulary vocabularies. In the sixth section we look at consequences of this result from a complexity theoretic point of view, and show any first-order formula can be evaluated in linear-time, regardless of its arity. The conclusion reexamines the questions which began this introduction, and highlights ideas behind scalable models of information.

2. Singulary Logic

According to the Oxford English Dictionary, the word *singulary* is used exclusively in logic, referring to mathematical operations “involving just one element”. The term was introduced by W.V. Quine, who preferred it to the more common usage ‘unary’ because of Latin derivation [12]. It became adopted by Alonzo Church who, in concurring with Quine on this etymological point, went so far as to define singulary vocabularies in which all the function and predicate symbols have only one place [1]. This constraint formally prohibits tupling, and singulary structures can be visualized as directed graphs of regular out-degree, where each node is the origin of exactly one type of outgoing edge for each function, and is colored by the combination of predicates that make it true.

Terminologically, we distinguish between ‘monadic’ and ‘unary’ when referring to intentional and extensional objects respectively, though we make no such distinction for the adjective ‘singulary’, using it in both situations. For a symbolic vocabulary to be singulary, all variables must appear alone as the object of predicate or function application (equality is excluded from this restriction so that one variable can appear on each side of an equals sign).

Definition 1 A singular vocabulary has only monadic predicates and functions:

$$L = \{P_1, \dots, P_m, f_1, \dots, f_k\} \quad \text{where each symbol has precisely one place.}$$

Singular vocabularies are interpreted by singular structures in the obvious way:

Definition 2 A singular structure S has only unary relations and functions:

$$S = \langle |S|, P_1, \dots, P_m, f_1, \dots, f_k \rangle \quad \text{where each } P_i \subseteq |S|, \text{ and each } f_j : |S| \rightarrow |S|.$$

Notice that the *total size* of S , which includes the number of tuples occurring in S , $\|S\| + |P_1| + \dots + |P_m| + |f_1| + \dots + |f_k|$, is $O(|S|)$. As is customary, we will refer to S as an L -structure.

A collection of L -structures is called an L -class. Classes over singular vocabularies are called *singular classes*. First-order formulas are defined in the usual way. As is customary, a first-order sentence over the vocabulary L will be referred to as an L -sentence, boolean queries which define properties of L -classes. The semantics of L -formulas are no different than those of formulas in ordinary first-order logic. However, it is entirely possible for a formula over a singular vocabulary to define a binary relation, such as $\varphi(x, y) \equiv x \neq y$. In this case, the formula φ is not itself singular. The term singular formula will be reserved for formulas with precisely one free variable.

It is important to realize that even though equality is technically a binary relation, it is not necessary to use pairs of elements for its interpretation. This is unlike the more general case of a graph where knowledge of its edges is required, and cannot be determined by an examination of its vertices alone. In other words, equality is not a real relation in the sense that it does not engage distinct elements of a structure. Yet its presence is required as we shall see in subsequent sections.

3. Singular Interpretations

This section will illustrate how structures in any vocabulary can be elementarily interpreted as singular structures. Every class of L -structures is uniformly transformed into a class of singular L^* -structures which mirrors the same first-order definable properties. Although this follows easily from well-known results, our construction is instructive nonetheless. A similar but slightly more complicated construction appears in [2].

It is obvious we can assume without loss of generality a class of purely relational structures in a vocabulary L whose relations are of common arity $k \geq 2$. Each such L -structure $S = \langle |S|, R_1, \dots, R_m \rangle$ of arity k will be mapped to a singular L^* -structure of size $O(|S|^k)$ by a uniform construction.

Definition 3 Let $S = \langle |S|, R_1, \dots, R_m \rangle$ be an L -structure of arity k . For the singular vocabulary $L^* = \{P_1, \dots, P_m, f_1, \dots, f_k\}$, define the L^* -structure:

$$S^* = (\overline{S} \cup R_1 \cup \dots \cup R_m, P_1, \dots, P_m, f_1, \dots, f_k) \quad \text{where } \overline{S} = \{\langle s, \dots, s \rangle \in |S|^k\}$$

$$P_i(\langle s_1, \dots, s_k \rangle) \text{ iff } S \models R_i(s_1, \dots, s_k) \quad \text{for } 1 \leq i \leq m$$

$$f_j(\langle s_1, \dots, s_k \rangle) = \langle s_j, \dots, s_j \rangle \quad \text{for } 1 \leq j \leq k$$

The domain of S^* is the set of k -tuples $\langle s_1, \dots, s_k \rangle$ appearing in S , either from a *fact* in some relation R_i or as *ground* elements $\langle s, \dots, s \rangle$ representing an original member of S . The predicates mark the nodes for each respective relation. The functions serve to project the nodes so that they can be interpreted as k -tuples – the j^{th} component of s is given by $f_j(s)$.

The simplest example is that of an ordinary directed graph, where $m = 1$ and $k = 2$. In figure 1, round vertices and straight edges are used to depict the original graph. For the singularly structure, functions f_1 and f_2 are drawn as curved arrows originating on the left and right respectively, and the predicate P_1 is demarcated by using boldface inside the rectangular nodes.

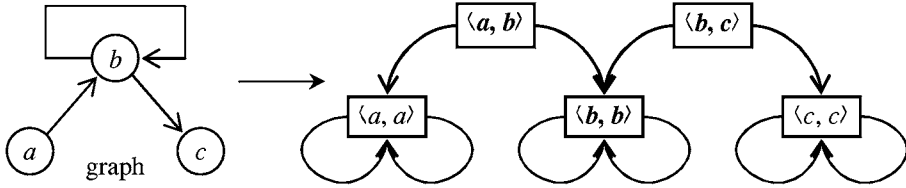


Fig. 1. Transforming a graph into a singularly structure.

Notice that this is more or less equivalent to passing from a graph to its incidence structure. The cardinality of a singularly structure accurately represents the amount of information it contains – one node for each fact. Assuming each fact necessitates some matter for storage, the size of a singularly structure is proportional to its total mass since, on a per node basis, each monadic predicate contributes one bit, and each monadic function contributes one element.

The proof of the following proposition is an easy consequence of the given construction, the details of which can be found in a previous version of this paper [11]. A similar result appears in [2], and in any event is easily derivable from the well-known fact that a vocabulary consisting of a single binary relation suffices (e.g. Exercise 3.7 of [8]).

Proposition 1 *Let $L = \{R_1, \dots, R_m\}$ be a relational vocabulary of arity $k \geq 2$. Then for each L -sentence θ there is an equivalent L^* -sentence θ^* such that for all L -structures, $S \models \theta \Leftrightarrow S^* \models \theta^*$. Moreover, the L^* -class $\{S^* : S \text{ is an } L\text{-structure}\}$ is an elementary class.*

Also notice that, to within a constant factor depending only on L , the total size of S as a mathematical structure, $|S| + |R_1| + \dots + |R_m|$, is the same as that of S^* , $|S^*| + |P_1| + \dots + |P_m| + |f_1| + \dots + |f_k|$.

4. Bounded-degree Classes

The previous section showed that with respect to elementary definability, singular models are no less general than arbitrary models. Moreover, passage to a singular model does not materially affect the total size, or amount of data. But in general, there will be an unbounded number of elements that reference the ground elements, leading to unbounded-degree. Since our real interest is in examining structures of bounded-degree, the aim of this section is to show that the total degree, or concentration of data, is not significantly affected when passing to a singular model.

The extent to which elements in an L -structure S are related can be determined by examining its Gaifman graph $G(S)$, a simple graph over the same domain which puts an edge between a pair of nodes iff they occur jointly in a tuple of some relation (see figure 2 for an example).

$$G(S) = \langle |S|, \{ \langle a, b \rangle : a \neq b \ \& \ S \models R(\dots a, \dots b, \dots) \vee R(\dots b, \dots a, \dots) \text{ for some } R \text{ in } L \} \rangle$$

The *degree* of an element in S is just defined to be its degree in the Gaifman graph $G(S)$. The degree of a structure S , $\text{deg}(S)$, is simply the supremum of degrees realized in S , and a class of structures C is said to be of bounded-degree if for every S in C , $\text{deg}(S)$ is less than some d .

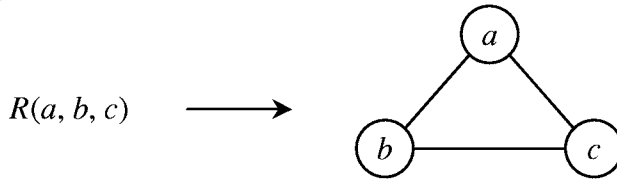


Fig. 2. A triangle of edges contributed to the Gaifman graph by a 3-tuple.

When measuring the total mathematical size of a structure S , it makes sense to include all the tuples $T(S)$ appearing in any relation (or function), since those tuples are the data being stored in any implementation. By assuming bounded proximity between a tuple and the elements it refers to, the local concentration of that data about an element s can be gauged by the number of tuples nearest s , in other words the number of tuples that s appears in. This motivates the following.

Definition 4 *In a relational L -structure S , the total degree of an element s is:*

$$|\{ \langle s_1, \dots, s_i, \dots, s_k \rangle \in T(S) : \text{some } s_i = s \}| \text{ where } T(\langle |S|, R_1, \dots, R_m \rangle) = R_1 \cup \dots \cup R_m.$$

It is straightforward to show that the total degree of s is polynomial in its ordinary degree, completely independent of the size of S (a proof can be found in [11]).

Fact: If s has degree d in a k -ary structure, then the total degree of s is $O(d^{k-1})$.

A singular model $M = \langle |M|, P_1, \dots, P_m, f_1, \dots, f_k \rangle$ can always be represented as a relational structure $M' = \langle |M|, P_1, \dots, P_m, R_1, \dots, R_k \rangle$, replacing each function f_i by the binary relation $R_i = \{ \langle a, b \rangle : f_i(a) = b \}$ representing its graph. The

Gaifman graph of M' will just be the symmetric irreflexive closure of $R_1 \cup \dots \cup R_k$. So in particular, the total degree of any element in M is at most linear in d , its ordinary degree.

All that remains to be observed is that the construction of S^* in the previous section preserves the total degree of elements in S , up to a constant factor, and doesn't introduce any new elements of unbounded degree. The former is obvious from the definition of the functions in S^* , and the latter since the degree of elements not in \bar{S} is less than or equal to k by construction. So we have,

Proposition 2 *Let C be a class of relational structures of arity k . Then C is of bounded-degree iff the class of singular structures $C^* = \{S^* : S \text{ in } C\}$ is also of bounded-degree.*

5. Singular Models for Doubly-Linked Data Structures

Data structures are standard models of how information is stored in a computer – finite arrangements of nodes containing data, each linked to one another by pointers. It provides a mechanism of dynamically increasing storage capacity, keeping the amount of information contained in each node fixed, thereby providing a uniform method of storing and accessing the data. Although the structure (pattern of links) can be arbitrary, the amount of Boolean data stored at each node is fixed, as well as the number of outgoing pointers. Each node can be visualized as a fixed-width container holding bits and addresses. It is easy to see that singular models are a nearly perfect match for data structures. Each bit corresponds to the Boolean value of a predicate, and each address to the value of a function (nil values being an exception, dealt with later).

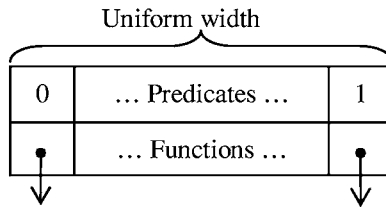


Fig. 3. A node in a data structure as an element in a singular model.

Extracting information from a data structure via an algorithm requires navigating the links. The easiest way to do this is when the structure is ‘doubly-linked’ – each link is provided with a return link, providing a convenient way to reverse direction and back out of a node. This also forces the data structure to be of bounded-degree, since the number of incoming links must not exceed the number of outgoing links, which are fixed. So clearly, the natural model for doubly-linked data structures is to use a singular vocabulary which has an inverse for each function. The details can be found in [11], and it is important to observe that the conversion process can be performed in linear-time.

Definition 5 An invertible singular vocabulary L has an inverse for each function:

$$L = \{P_1, \dots, P_m, f_1, \dots, f_k, f_1^\circ, \dots, f_k^\circ\}.$$

We write f° to denote the syntactic inverse of f , in order to distinguish it from the semantic inverse, and to indicate that it is a separately interpreted symbol (which happens to always be the inverse). When viewed as a graph, each bijective pair (f, f°) determines edges which partition the finite domain into a collection of disjoint cycles.

A problem seems to arise in the case of nil values, appearing to require partial functions, or introducing an element \perp with an arbitrarily large number of incoming arrows. But these are easily dealt with by *wraparound* as illustrated in figure 4, entirely obviating the need for \perp .

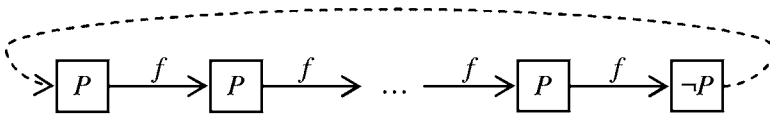


Fig. 4. Illustration of wraparound.

The dotted line represents a value introduced to replace nil, ensuring totality for f . It is determined uniquely by tracing backwards along the value chain for f (always possible since f is injective). This procedure terminates because every data structure has a finite domain – the final value realized is the one desired. The only thing left is to introduce a new predicate symbol P whose sole purpose is to indicate the endpoints where dotted lines were employed. This enables recovery of the original partial function via $f(x) = \perp$ iff $\neg P(x)$.

The real value of viewing doubly-linked data structures as singular models will become apparent in the next section, where the mathematical elegance of these invertible vocabularies bears fruit by allowing first-order formulas to collapse into a much simpler form.

6. Normal Forms

Invertible singular structures are not only regular, but elementary definability takes on a particular elegance – first-order formulas can always be written in a special normal form where quantifiers are never nested. To demonstrate this we introduce a special type of threshold quantifier which generalizes in a natural way ordinary existential quantification.

Definition 6 For each natural number n , the numerical quantifier $\exists^n x \dots$ stands for “there are (at least) n distinct x ’s satisfying ...”. In case $n = 1$, this is the ordinary existential quantifier.

Obviously, numerical quantifiers do not extend the realm of first-order definability because for any fixed n , \exists^n can be rewritten using n ordinary existential quantifiers.

$$\exists^n x \omega(x) \equiv \exists x_1 \dots x_n \{x_i \neq x_j : 1 \leq i < j \leq n\} \wedge \{\omega(x_l) : 1 \leq l \leq n\}$$

But they do serve to simplify our normal form theorem, which says that over any class of invertible singular structures (finite or infinite), every first-order formula can be rewritten as a Boolean combination of quantifier-free formulas together with sentences composed of numerical quantifiers applied to quantifier-free formulas. In particular, the quantifiers are never nested.

Theorem 1 *Let $\zeta(\bar{x})$ be a first-order formula in an invertible singular vocabulary. Then $\zeta(\bar{x})$ is equivalent to a Boolean combination of atomic formulas $\alpha(\bar{x})$ and numerically quantified sentences $\exists^n x \beta(x)$ where β is quantifier-free.*

Proof. By induction on the ordinary first-order syntax – all cases except quantification are trivial. Consider $\exists x \varphi(x, \bar{y})$ where φ is in the normal form as specified in the statement of the theorem. By putting φ in disjunctive normal form, distributing the existential quantifier over the disjunctions, and removing any sentences from underneath the scope of quantification, it suffices without loss of generality to consider φ as a conjunction of atomic clauses (or their negations). In a singular environment, clauses involving a predicate symbol contain only one variable. So the only way that variables can be related to each other is through the equality symbol. Each equation relates just two variables (not necessarily distinct), one occurrence on each side of the equals sign.

If a variable has no connection to x via equations in φ , it should be removed. Say that two (distinct) variables are *related* if they occur jointly in an equation together, and *connected* if they are in the (reflexive) transitive closure of this symmetric relation (i.e. directly or indirectly related). All variables not connected to x can be factored out from under the scope of the quantifier $\exists x$. To see how, take any quantifier-free conjunctive formula $\theta(x, \bar{y}, \bar{z})$ in which x is connected to every y in \bar{y} and no z in \bar{z} . Since it is impossible for any z to occur in the same clause with x or any y , we can simply separate out all clauses involving variables from among the \bar{z} as follows:

$$\exists x \theta(x, \bar{y}, \bar{z}) \quad \text{becomes} \quad \phi(\bar{z}) \wedge \exists x \varphi(x, \bar{y})$$

So without loss of generality, we can assume that each y in $\varphi(x, \bar{y})$ is connected to x . If there are no \bar{y} , we are done (for the sentence $\exists x \varphi(x)$ is already in the correct form). Otherwise x must participate in at least one nontrivial equation (i.e. an equation involving another variable y in \bar{y}), which must be of the form $G(x) = H(y)$, where G and H are compositions of the function symbols. These can always be solved for x by utilizing the critical assumption of bijectivity to rearrange the terms:

$$G(x) = H(y) \quad \text{becomes} \quad x = G^\circ H(y)$$

where G° stands for the inverse sequence of functions appearing in G , a crucial step that is only possible in an invertible singular vocabulary. So assume without loss of generality that whenever x appears in a nontrivial equation from φ , it is already in the form $x = F(y)$ or $x \neq F(y)$, where F stands for any composition of functions. A positive equation $x = F(y)$ is called an *identity*, whereas a negative

equation $x \neq F(y)$ will be called a *diversity*. After solving for x , there are two cases depending on the polarity of its equations.

Case 1: If an identity $x = F(y)$ appears in φ for some y in \bar{y} , just substitute the term $F(y)$ for x everywhere and eliminate the quantifier:

$$\exists x \varphi(x, \bar{y}) \quad \text{becomes} \quad \varphi(F(y), \bar{y})$$

where $\varphi(F(y), \bar{y})$ is the formula resulting from substituting $F(y)$ for x everywhere in $\varphi(x, \bar{y})$. Observe that the resulting formula is of the correct form (i.e. it is quantifier-free). Clearly, if φ asserts that $x = F(y)$, then $\exists x \varphi(x, \bar{y})$ can only be true provided $\varphi(F(y), \bar{y})$. On the other hand, if $\varphi(F(y), \bar{y})$ is satisfied, then $\exists x \varphi(x, \bar{y})$ is satisfied by $x = F(y)$.

Case 2: Otherwise, every nontrivial equation in φ involving x must be a diversity $x \neq F(y)$. The remaining occurrences of x in φ must be single variable occurrences. These properties of x fall into two types of atomic forms: positive $P(F(x))$ or negative $\neg P(F(x))$ predicates; and trivial equations $x = F(x)$ or $x \neq F(x)$ representing closed or open paths from x respectively. No matter what form they take, collect all these properties together into a formula $\gamma(x) \equiv \bigwedge \{\alpha(x) : \alpha(x) \text{ is a clause in } \varphi(x, \bar{y})\}$, and rewrite φ :

$$\varphi(x, \bar{y}) \quad \text{becomes} \quad \gamma(x) \wedge \psi(x, \bar{y})$$

where the only clauses in ψ involving x are diversities $x \neq F(y)$. As observed previously, x must relate to at least one y in \bar{y} . In order for x to exist, it must both satisfy γ and be distinct from any possible ‘‘competitors’’ $F(y)$ which may happen to also satisfy γ . E.g., if none of the $F(y)$ satisfy γ , then we are done, as $\exists x [\gamma(x) \wedge \psi(x, \bar{y})]$ is equivalent to $[\exists x \gamma(x)] \wedge \psi'(\bar{y})$, where ψ' is $\psi(x, \bar{y})$ in which all appearances involving x (i.e. diversities) have been deleted. More generally, to take into account all possibilities, consider any subset T of terms $F(y)$ diverse from x (i.e. x cannot be coincident with any member of T):

$$T \subseteq \{F(y) : x \neq F(y) \text{ is in } \psi\} \quad (*)$$

No term in T can be in competition with x unless it satisfies γ . So let the formula $\gamma[T] \equiv \bigwedge \{\gamma(F(y)) : F(y) \in T\}$ say that each of these terms satisfy γ (note $\gamma[\emptyset]$ is vacuously true). The size of T is a reliable indicator of the number of possible competitors only if the terms in T are distinct from one another. So let $\delta[T] \equiv \bigwedge \{F(y) \neq F'(y') : F(y) \ \& \ F'(y') \text{ are distinct terms in } T\}$ be the formula that says these terms are distinct (again, $\delta[\emptyset]$ is vacuously true). Claim the following conjunction is equivalent to the original $\exists x \varphi(x, \bar{y})$.

$$\zeta(\bar{y}) \equiv \bigwedge \{\delta[T] \wedge \gamma[T] \rightarrow (\exists^{n+1} z) \gamma(z) : \text{where } n = |T|, \text{ and } T \text{ ranges over } (*)\}.$$

This says that there are more elements satisfying γ than the number of distinct terms in T that also satisfy γ . Clearly, ζ is of the correct form (a combination of quantifier-free formulas or numerical quantifier sentences). It remains to be seen that $\zeta(\bar{y})$ is semantically equivalent to $\exists x \varphi(x, \bar{y})$. But this is pretty obvious. If

$\gamma(x) \wedge \psi(x, \bar{y})$ is satisfied by some element x in a situation where the n distinct elements $F(y) \neq x$ of T also satisfy γ , then there are indeed more than n elements satisfying γ . On the other hand, ζ says that every possible collection of n distinct elements $F(y) \neq x$ satisfying γ always leaves room for another element (namely x) which satisfies γ . \square

Note: An alternate method is to apply Gaifman’s Theorem in a relational setting to get a Boolean combination of local formulas and sentences which say there are n disjoint neighborhoods satisfying the same quantifier-free formula. In our singular setting, the former correspond to quantifier-free formulas and the latter to numerical sentences which will not yield to linear-time model checking (as explained in the next section). However, if we do not mind a semantic argument, applying the bounded-degree version of Hanf’s Lemma to these sentences will result in the correct sentential form.

7. Linear-time Computability

An immediate consequence of the normal form theorem is a linear-time algorithm for evaluating any first-order sentence in an invertible singular vocabulary, essentially the same as the pioneering result of [13] over bounded-degree graphs. We improve upon this by extending its reach to formulas in a unique way, attempting to give a convincing analysis that not only sentences, but formulas with any number of free variables, can be realistically evaluated in linear-time, despite apparent contradictions in output size.

Ordinarily, evaluating a first-order formula $\sigma(\bar{x})$ over a finite model M involves explicitly computing the tuples determined by $\sigma^M = \{\bar{m} : M \models \sigma(\bar{m})\}$. When $\sigma(\bar{x})$ has multiple free variables we cannot hope to compute the entire relation determined by σ^M in linear-time because the number of tuples is potentially polynomial in the size of M , with degree the arity of \bar{x} . A solution around this problem is to compute a *partial interpretation* $\sigma^M(\bar{x})$ which, when given values \bar{m} for the free variables, can answer in constant-time whether or not $M \models \sigma(\bar{m})$.

Definition 7 An L -query $q(\bar{x})$ is said to be implicitly computable in linear-time if for every finite model M , there is a partial interpretation $q^M(\bar{x})$ such that:

- (i) Given M , $q^M(\bar{x})$ can be computed in $O(|M|)$ time
- (ii) Given \bar{m} , $q^M(\bar{m}) \Leftrightarrow M \models q(\bar{m})$ can be computed in $O(1)$ time

For a formula $\sigma(\bar{x})$ this means that for each M , we compute in time linear in M another formula $\sigma^M(\bar{x})$ which demonstrates that we implicitly know which tuples satisfy σ over M even though we cannot list them explicitly (no different than the trick used to compose space-bounded algorithms in which the intermediate result exceeds the allotted bound), so the arity will not constrain the efficiency of its evaluation as illustrated in figure 5.

Corollary 1 Any L -formula $\phi(\bar{x})$ in an invertible singular vocabulary is implicitly computable in linear-time.

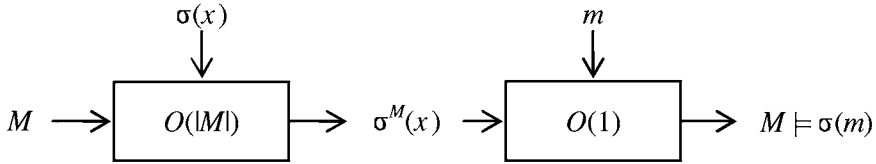


Fig. 5. Method of partial interpretations used to implicitly calculate a query.

Proof. By the main theorem, $\phi(\bar{x})$ can be written in normal form as a Boolean combination of quantifier-free formulas, and sentences which must quantify a single variable x over formulas necessarily limited to being composed of properties of paths, $P(F(x))$, or cycles, $x = F(x)$, from x . Given an L -structure S , each of the constituent sentences $\exists^k x \beta(x)$ can be explicitly evaluated in linear-time by simply scanning each node s of S to see if it satisfies $\beta(s)$. Checking $\beta(s)$ involves following bounded-length paths from s of the form $F(s)$ to see if they are closed or open (equal or not equal to s), and possibly checking a predicate P there. This requires $O(1)$ time for each s , together with counting the results. By substituting those Boolean valuations into the original formula $\phi(\bar{x})$, the resultant quantifier-free formula $\phi^S(\bar{x})$ is the desired partial interpretation. Given values \bar{s} for its free variables, ϕ^S requires only $O(1)$ time to answer whether or not $\phi^S(\bar{s})$, which is obviously equivalent to $S \models \phi(\bar{s})$. \square

Note: In order to apply this result to a doubly-linked data structure, first perform the linear-time conversion to an invertible singularly structure as outlined in §4.

8. Conclusion

Ordinary singularly structures (the non-invertible variety) appear to provide an answer to the first question that began this paper. We have seen that all relational structures can be uniformly transformed into simple types of structures which do not utilize tuples – their predicates and functions are unary – while retaining all elementarily definable properties. Moreover, this transformation proportionally preserves the total size of the original structure, and the total degree of the elements inside, both measured in a very natural way. Not only do we have a complete answer to the first question of the introduction, but one which does not disturb fundamental notions of cardinality and clustering.

A full answer to the second question posed in the introduction has not been provided. It would be necessary to go into a detailed analysis of physical information storage, and the interested reader is encouraged to look at [10] for a more complete investigation. But for this paper it suffices to realize that a scalable model of information must be based on data structures containing datum with uniform capacity, a notion that is *sine qua non* to the whole idea of a class of structures being asymptotically physically realizable. We saw that singularly models correspond naturally to data structures of this type – containing nodes of a fixed width. If stored in physical memory, these nodes would necessarily occupy material locations in space and it would be hard to imagine why the actual links between them could

not be made reciprocal. Invertible singularly models are a nearly ideal match to these doubly-linked data structures.

Our main theorem shows that under these circumstances, first-order definability is much simpler than it appears. In particular, quantifiers are never applied to formulas with more than one variable, permitting a very straightforward linear-time evaluation algorithm. For future research, perhaps similarly strong normal forms can be obtained for monadic fixed-point logic over invertible singularly vocabularies.

Acknowledgments

Help in preparing this manuscript for publication came from my wife, Suzanne Lindell, funded by Haverford College. Her assistance and support from my college is greatly appreciated. I also wish to thank the Newton Institute for Mathematical Sciences for their support in visiting Cambridge, England to give a lecture based on this paper. This research was also funded by an NSF SGER grant, #CCR-0225063. The author would also like to thank several anonymous referees for their useful advice and suggestions.

References

1. A. Church, *Introduction to Mathematical Logic*, Princeton University Press, 1956.
2. A. Durand & E. Grandjean, 'First-order queries on structures of bounded degree are computable with constant delay' *ACM Trans. on Computational Logic*, 2007 to appear.
3. H. Gaifman, 'On local and non-local properties' in *Proceedings of the Herbrand Symposium: Logic Colloquium '81*, (J. Stern ed.) pp. 105-135 (North-Holland, Amsterdam 1982).
4. E. Grandjean, 'Invariance properties of RAMs and linear time' *Computational Complexity*, 4:62-106, 1994.
5. E. Grandjean, 'Linear time algorithms and NP-complete problems' *SIAM J. on Computing*, 23(3):573-597, 1994.
6. E. Grandjean and T. Schwentick, 'Machine-independent characterizations and complete problems for deterministic linear time' *SIAM J. on Computing*, 32(1):196-230, 2002.
7. W. Hanf, 'Model-theoretic methods in the study of elementary logic' in *The Theory of Models* (J. Addison, L. Henkin and A. Tarski, eds.) pp. 132-145 (North Holland, Amsterdam 1965).
8. N. Immerman, *Descriptive Complexity*, Springer, New York 1999.
9. L. Libkin, *Elements of Finite Model Theory*, Springer, 2005.
10. S. Lindell, 'An elementary term logic for physically realizable models of information', in *The Old New Logic: Essays on the Philosophy of Fred Sommers*, David Oderberg editor, MIT Press, 2005.
11. S. Lindell, 'A normal form for first-order logic over doubly-linked data structures' *Isaac Newton Institute Preprint Series*, 2006.
12. W.V. Quine, *Mathematical Logic*, W.W. Norton & company, 1940.
13. D. Seese, 'Linear time computable problems and first-order descriptions' in *Mathematical Structures in Computer Science*, 6(6):505-526, December 1996.

Copyright of International Journal of Foundations of Computer Science is the property of World Scientific Publishing Company and its content may not be copied or emailed to multiple sites or posted to a listserv without the copyright holder's express written permission. However, users may print, download, or email articles for individual use.