JSL Review

S. R. Buss

March 6, 2002

N. Immerman. *Upper and lower bounds for first order expressibility.* **Journal of Computer and System Sciences** vol. 25 (1982) 76-98.

N. Immerman. *Relational queries computable in polynomial time.* **Information and Control** vol. 68 (1986) 86-104.

N. Immerman. *Languages that capture complexity classes.* **SIAM Journal on Computing** vol. 16 (1987) 760-778.

Given a relational language $\mathcal{L}$ and a class of finite structures for $\mathcal{L}$, it is a natural question to ask for ways of characterizing the class. The papers under review treat this problem from the viewpoint of computer science by discussing the relationship between expressibility of properties in various extensions of first order logic and the computational complexity of determining if a given structure satisfies such a property. The primary languages considered are formed by extending first order logic with a least fixed point operator or with a transitive closure operator or by allowing "infinitary" formulas. The primary classes of computational complexity are NL (nondeterministic log space), P (polynomial time) and *PSPACE* (polynomial space).

First order logic alone is too weak to express many interesting properties of finite structures; for example, there is no first order formula expressing the graph property connectivity. In *Upper and lower bounds for first order expressibility,* Immerman proposes that an infinite sequence of first order formulas $\varphi_1, \varphi_2, \ldots$ be used to define the class $\mathcal{C}$ of finite structures such that for any $\mathfrak{A}$ of cardinality n, $\mathfrak{A}$ is in $\mathcal{C}$ iff $\mathfrak{A} \models \varphi_n$. Of course any class of finite structures can be defined in this way; so in order to get interesting results, Immerman restricts the size of and the number of distinct variables in φ_n. The following (slightly modified) example of Immerman's illustrates most of the principles: Graphs are structures in the language with one binary relation symbol E satisfying $\forall x, y[\neg E(x,x) \wedge (E(x,y) \rightarrow E(y,x))]$. If n is even, the formula $P_n(x,y)$ which asserts that there is a path of length $\leq n$ from vertex x to vertex y can be defined recursively as $\exists z(P_{n/2}(x,z) \wedge P_{n/2}(z,y))$. This definition gives P_n size $O(n)$; a better definition is $P_n(x,y) \equiv (\exists z)(\forall v \cdot v = x \vee v = y)(\exists x \cdot x = z)(\exists y \cdot y = v)P_{n/2}(x,y)$ where $(Qu \cdot M)$ is quantification restricted to values satisfying M. So for graph connectivity we can let φ_n be $\forall x \forall y P_n(x,y)$ and each φ_n has 4 distinct variables and size $O(\log n)$. Indeed, φ_n can be written as $\forall x \forall y [QB]^{\lceil \log n \rceil}(E(x,y) \vee x = y)$ where $[QB]^m$ denotes the above quantifier block iterated m times.

A class of structures is $Var\&Sz[v(n), z(n)]$ if is it definable by a family of first order formulas $\varphi_1, \varphi_2, \ldots$ such that φ_n has $v(n)$ distinct variables and

has size $O(z(n))$. Furthermore, the φ_n's must be suitably uniform: for $v(n)$ a constant function, φ_n must be definable by an iterated quantifier block as in the above example; otherwise it is required that φ_n can be generated by a Turing machine which uses space $v(n)\log n$ and time $z(n)$.

A finite structure can be described in a canonical way by a string from a finite alphabet; thus the computational complexity of a class $\mathcal{C}$ of structures is defined to be the computational complexity of the set of strings which represent members of the class. To get relationships between the $Var\&Sz$ classes and computational complexity, it is necessary to require that the language $\mathcal{L}$ contain a binary relation symbol $\leq$ which is always a total ordering. With this assumption, Immerman shows that if $s(n) \geq \log n$, then $ASPACE\&TIME[s,t] \subseteq Var\&Sz(s/\log n, t) \subseteq ASPACE\&TIME[s, t\log n]$; and concludes the the number of variables corresponds closely to alternating space and the size to alternating time. Also the classes P and $PSPACE$ can be characterized by $P = \bigcup_k Var\&Sz[k, n^k]$ and $PSPACE = Var\&Sz[k, 2^{n^k}]$.

$Log(CFL)$ is the class of predicates which are many-one, log space reducible to a context free language. Using work of Sudborough and Ruzzo, it is shown that $Log(CFL)$ is $Var\&Sz(B\forall)[O(1), \log n]$ where the $(B\forall)$ indicates the the universal quantifiers must be Boolean.

The use of $Var\&Sz[-,-]$ classes is closely related to the use of uniform families of circuits to recognize predicates. For example, Immerman has borrowed much from earlier work on circuit complexity; on the other hand, Venkateswaran in *Properties that characterize LOGCFL*, **19th ACM Symp. on Theory of Computing**, 1987, pp. 141-150, has recast Immerman's characterization of $Log(CFL)$ in terms of semiunbounded fanin circuits. One advantage to using first order formulas instead of uniform Boolean circuits is that when only a constant number of variables are allowed, the uniformity condition on the first order formulas is very elegant and does not depend on Turing machine complexity.

In *Relational queries computable in polynomial time*, the effect of adding a least fixed point operator to first order logic is studied. If R is a k-ary predicate and φ is a formula with k free variables in which R appears only positively then $(\mu R)\varphi$ denotes the least predicate R such that $\forall\vec{x}(R(\vec{x}) \leftrightarrow \varphi)$ holds; $FO + LFP$ denotes first order logic plus this operator. For example the graph property $P(x,y)$ that there is a path from x to y can be defined as $(\mu R)[x = y \vee \exists z(R(x,z) \vee E(z,y))]$.

There are two main results concerning the LFP operator. Firstly, if there is a binary relation $\leq$ which is always a total ordering then a predicate is expressible in $FO{+}LFP$ iff it is in P; this was independently proved by M. Vardi. Secondly, even without $\leq$ in the language, every formula of $FO{+}LFP$ can be expressed as the least fixed point of a first order formula, i.e., as $(\mu R)\varphi$ where φ is first order. In particular, the least fixed point operation does not need to be nested; this collapses a hierarchy defined by Chandra and Harel. The latter result is proved using the machinery of Moschovakis, **Elementary Induction on Abstract Structures**, North-Holland, 1974, although the results are different for finite and infinite structures. Gurevich and Shelah later showed that the same results

hold when the definition of least fixed point is broadened to allow $(\mu R)\varphi$ to be definable whenever $\varphi(R)$ is monotone in R.

The second paper concludes by showing that the least fixed point operator may be translated directly into an iterated quantifier block, so any $FO + LFP$ formula defines a $Var\&Sz[O(1), n^{O(1)}]$ class of structures.

In *Languages that capture complexity classes*, a transitive closure operator TC is added to first order logic; if $\varphi(\vec{x}, \vec{y})$ is viewed as a binary relation on k-tuples then $TC[\varphi(\vec{x}, \vec{y})]$ denotes the transitive closure of φ. Let $FO + \mathrm{pos}\, TC$ denote the properties expressible with first order logic plus positive occurrences of the TC operator. In this paper, instead of assuming that $\leq$ is in the language, it is required that there is a successor relation s and constants 0 and `max` so that the entire universe can be enumerated by taking successors of 0. Then any $FO + \mathrm{pos}\, TC$ formula is equivalent to the transitive closure of a first order formula and $FO + \mathrm{pos}\, TC = NL$. Similar results hold for deterministic and symmetric transitive closure operators DTC and STC; in particular, $FO + DTC$ is log space and $FO + \mathrm{pos}\, STC$ is symmetric log space.

There has been recent significant progress on the TC operator: in *Descriptive and computational complexity* in **AMS Short Course on Computational Complexity**, January 5-6, 1988, Atlanta, Georgia, Immerman has shown that $FO + TC = FO + \mathrm{pos}\, TC$. It follows immediately that $FO + TC = NL$ and that arbitrary formulas involving TC may be replaced by a single transitive closure of a first order formula. Even more significantly, it follows that NL is closed under complementation and that the class of context sensitive languages is closed under complementation. These latter results resolve a long standing open question and were independently obtained by R. Szelepcsényi, *The method of forcing for nondeterministic automata*, **Bulletin of the European Association for Theoretical Computer Science** vol. 33 (1987) pp. 96-99. This also refutes Immerman's conjecture 3(a) that a transitive closure hierarchy is proper; although his conjecture 3(c) regarding the transitive closure operator without the successor relation is still open.

The papers under review are well written and motivated and generally complete and accurate. The subject area, definability in finite structures, is likely to remain an important area of research connecting mathematical logic and theoretical computer science.

S. Buss