A note on first-order projections and games

Argimiro A. Arratia[*]
Departamento de Matemáticas, Universidad Simón Bolívar,
Apartado 89000, Caracas 1080-A, Venezuela

Iain A. Stewart[†]
Department of Mathematics and Computer Science,
University of Leicester, Leicester LE1 7RH, U.K.

Abstract

We show how the fact that there is a first-order projection with successor from the problem TC (transitive closure) to some other problem Ω enables us to automatically deduce that a natural game problem, $\mathcal{LG}(\Omega)$, whose instances are labelled instances of Ω, is complete for **PSPACE** (via log-space reductions). Our analysis is strongly dependent upon the reduction from TC to Ω being a logical projection in that it fails should the reduction be, for example, a log-space reduction or a quantifier-free first-order translation with successor.

1 Introduction

In this note, we show how the existence of a first-order projection with successor (a very restricted type of logical reduction) from the problem TC (transitive closure) to some other problem Ω enables one to *automatically* deduce that a natural game problem, $\mathcal{LG}(\Omega)$, whose instances are labelled instances of Ω, is complete for **PSPACE** (via log-space reductions). Many problems have been shown to be complete for **NL**, and for complexity classes containing **NL**, via first-order projections with successor, and so there is a wealth of contenders for the problem Ω (as TC is in **NL**). It turns out that the notion of a first-order projection with successor is *exactly* the notion we require for our main result to hold and for it to be widely applicable. Completeness via stronger reductions, such as log-space or polynomial-time reductions, or other logical reductions, such as quantifier-free first-order translations with successor, does not suffice.

[*]Supported by EPSRC Grant GR/M 91006.
[†]Supported by EPSRC Grants GR/M 91006 and GR/M 12933.

2 Logical reductions

We begin with some definitions. The reader is referred to [1, 4] for more details. A *signature* σ is a tuple $\langle R_1, \ldots, R_r, C_1, \ldots, C_c \rangle$, where each R_i is a relation symbol, of arity $a_i > 0$, and each C_j is a constant symbol. A *finite structure $\mathcal{A}$ over the signature* σ, or *σ-structure*, consists of a finite *universe* or *domain* $|\mathcal{A}|$ together with a relation R_i of arity a_i, for every relation symbol R_i, and a constant $C_j \in |\mathcal{A}|$, for every constant symbol C_j (we do not distinguish between relations and relation symbols and between constants and constant symbols). A finite structure $\mathcal{A}$ whose domain consists of n distinct elements has *size* n, and we denote the size of $\mathcal{A}$ by $|\mathcal{A}|$ also (this does not cause confusion). We only ever consider finite structures of size at least 2, and the class of all finite structures of size at least 2 over the signature σ is denoted STRUCT(σ). A *problem* over some signature σ consists of a sub-class of STRUCT(σ) that is closed under isomorphism; that is, if $\mathcal{A}$ is in the problem then so is every isomorphic copy of $\mathcal{A}$. Throughout, all our structures are finite.

Immerman [3] obtained some very strong complexity-theoretic completeness results where the notion of reduction was a first-order projection with successor. Rather than give a general definition of a first-order projection with successor, we shall explain what it means for such a reduction to exist between two specific problems. The general definition can be found in detail in [4].

Let the signature σ_{2++} consist of the binary relation symbol E and the two constant symbols C and D. We can think of a σ_{2++}-structure as a digraph with two designated vertices (which may be identical). The problem TC consists of all those σ_{2++}-structures with the property that there is a path in the digraph from the vertex C to the vertex D.

Let the signature $\sigma_{2,2}$ consist of two binary relation symbols P and N. We can think of a $\sigma_{2,2}$-structure $\mathcal{A}$ as a collection of clauses of Boolean literals as follows. There is a Boolean variable X_u and a clause C_u, for every element $u \in |\mathcal{A}|$. The literal X_u is in the clause C_v if, and only if, $P(u, v)$ holds in $\mathcal{A}$; and the literal $\neg X_u$ is in the clause C_v if, and only if, $N(u, v)$ holds in $\mathcal{A}$. The problem SAT consists of all those $\sigma_{2,2}$-structures with the property that there is a truth assignment on the Boolean variables such that at least one literal in every non-empty clause is set at true.

Suppose that there exists a positive integer k and two first-order formulae φ_P and φ_N over the signature σ_{2++} such that the free variables of φ_P and φ_N are amongst the two k-tuples of variables $\mathbf{x}$ and $\mathbf{y}$ (all variables are distinct). Let $\mathcal{A}$ be a σ_{2++}-structure of size n. We shall use the formulae φ_P and φ_N to build a $\sigma_{2,2}$-structure $\rho(\mathcal{A})$. The domain of $\rho(\mathcal{A})$ is $|\mathcal{A}|^k$ and for every $\mathbf{u}, \mathbf{v} \in |\mathcal{A}|^k$, the literal $X_{\mathbf{u}}$ (resp. $\neg X_{\mathbf{u}}$) is in the clause $C_{\mathbf{v}}$ if, and only if, $\varphi_P(\mathbf{u}, \mathbf{v})$ (resp. $\varphi_N(\mathbf{u}, \mathbf{v})$) holds in $\mathcal{A}$. If it is the case that $\mathcal{A} \in$ TC if, and only if, $\rho(\mathcal{A}) \in$ SAT then we say that there is a *first-order translation* from TC to SAT.

It turns out that first-order translations are too restricted to describe many complexity-theoretic reductions. However, this difficulty can often be alleviated by augmenting such translations with a built-in successor relation. Let *succ* be a binary relation symbol and let 0 and *max* be two constant symbols (we assume that *succ*, 0 and *max* do not appear in any signature). Suppose that our formulae φ_P and φ_N, above, are actually over the signature $\sigma_{2++} \cup \langle succ, 0, max \rangle$ and let $\mathcal{A}$ be a

σ_{2++}-structure of size n. Denote by $S(\mathcal{A})$ the class of expansions of $\mathcal{A}$ by a successor relation *succ* with minimal element 0 and maximal element *max*; that is, by a relation *succ* of the form

$$\{(0,u_1),(u_1,u_2),\ldots,(u_{n-2},max)\},$$

where $0,u_1,u_2,\ldots,u_{n-2},max$ are all distinct. Clearly, by proceeding as above, any structure $(\mathcal{A},succ,0,max)$ in $S(\mathcal{A})$ yields a $\sigma_{2,2}$-structure $\rho(\mathcal{A},succ,0,max)$. If it is the case that for every σ_{2++}-structure $\mathcal{A}$ and for every expansion $(\mathcal{A},succ,0,max)\in S(\mathcal{A})$,

- $\mathcal{A}\in$ TC implies that $\rho(\mathcal{A},succ,0,max)\in$ SAT

and

- $\mathcal{A}\notin$ TC implies that $\rho(\mathcal{A},succ,0,max)\notin$ SAT

then we say that there is a *first-order translation with successor* from TC to SAT.

Whilst first-order translations with successor can be used to describe many complexity-theoretic reductions, we can often get by with restrictions of first-order translations with successor (but not so that we dispense with a built-in successor relation). Suppose further that the formulae φ_P and φ_N are both of the form

$$\alpha_0\vee(\alpha_1\wedge\beta_1)\vee(\alpha_2\wedge\beta_2)\vee\ldots\vee(\alpha_m\wedge\beta_m),$$

for some $m\geq 0$ (depending upon φ_P or φ_N), where: each α_i is a first-order formula over the signature $\langle C,D\rangle\cup\langle succ,0,max\rangle$; and each β_i is an atomic or negated atomic formula involving the relation symbol E. If it is the case that for every $\mathcal{A}\in$ STRUCT(σ_{2++}) and for both φ_P and φ_N, the formulae $\alpha_0,\alpha_1,\ldots,\alpha_m$ are mutually exclusive, i.e., for every $\mathbf{u},\mathbf{v}\in|\mathcal{A}|^k$, at most one of $\alpha_0(\mathbf{u},\mathbf{v}),\alpha_1(\mathbf{u},\mathbf{v}),\ldots,\alpha_m(\mathbf{u},\mathbf{v})$ holds in $\mathcal{A}$, then we say that there is a *first-order projection with successor* from TC to SAT.

We should add that in the general case, when a first-order projection with successor from a problem over some signature σ to a problem over some signature σ' is such that the 'target' σ'-structure (a $\sigma_{2,2}$-structure in the illustration above) involves constants then we diverge from the usual definition (as given in [4], for example). In order to describe a constant of σ', we insist that the describing formula be a first-order formula $\psi(\mathbf{x})$, where $\mathbf{x}$ is a k-tuple of variables (following in the vein of our illustration above), where there is exactly one k-tuple $\mathbf{u}\in|\mathcal{A}|^k$ for which $\psi(\mathbf{u})$ holds in $\mathcal{A}$ and where ψ involves no relation symbols of σ. This slight modification will be important later on.

First-order projections with successor are so called for the following reason. Fix $n\geq 2$, a domain U of size n, two elements C and D of U (possibly identical) and a successor relation *succ* (with minimal element 0 and maximal element *max*) on this domain. Consider the expansion $(\mathcal{A},succ,0,max)$ of the σ_{2++}-structure $\mathcal{A}$ with domain U and with constants C and D. Let us focus on some literal $\neg X_{\mathbf{u}}$ and clause $C_{\mathbf{v}}$, say, of the $\sigma_{2,2}$-structure $\rho(\mathcal{A},succ,0,max)$ (with the notation as above). As to whether the literal $\neg X_{\mathbf{u}}$ appears in the clause $C_{\mathbf{v}}$ depends upon two things: whether some α_i in φ_N is such that $\alpha_i(\mathbf{u},\mathbf{v})$ holds (a fact independent of $\mathcal{A}$, in so far as n, C, D and *succ* have been fixed); and if so then whether $\beta_i(\mathbf{u},\mathbf{v})$ holds, if $i>0$. Note that whether $\beta_i(\mathbf{u},\mathbf{v})$ holds depends upon at most one 'bit' of $\mathcal{A}$. Roughly, as to whether a 'bit' of $\rho(\mathcal{A}, succ, 0, max)$ is 'on' or 'off' depends upon whether at most one 'bit' of $(\mathcal{A}, succ, 0, max)$ is 'on' or 'off'; hence the name 'projection'.

Immerman introduced first-order projections with successor in [3] where he exhibited complete problems for the complexity classes **L**, **NL** and **P** via such reductions (in fact, via quantifier-free projections with successor). Since then, a number of other such logical completeness results have been obtained (see, for example, [6, 7]). However, it should be noted that although the built-in successor can be dispensed with in some logical completeness results for **NP** (see, for example, [7]), no completeness results via logical reductions without a built-in successor are known for complexity classes 'contained in' **NP**, like **L**, **NL** and **P**. Hence, we phrase our results in terms of first-order projections with successor for wider applicability.

We have one final remark. Even though we have restricted the usual definition of a first-order projection with successor (in comparison with Immerman's [4]), it turns out that many of the concrete first-order projections with successor from the literature are actually first-order projections with successor according to our definition. For example, the problem TC is complete for **NL** via our first-order projections (see [3]) and all the first-order projections with successor in [6, 7] conform to our definition.

3 Labelled structures and games

We begin by explaining how we label structures and how we play games on these labelled structures. In some σ-structure $\mathcal{A}$, if R is a relation symbol of σ of arity a and $R(u_1, u_2, \ldots, u_a)$ holds, for some $u_1, u_2, \ldots, u_a \in |\mathcal{A}|$, then we say that $R(u_1, u_2, \ldots, u_a)$ is an *instantiation* of $\mathcal{A}$.

Definition 1 Let $\mathcal{A}$ be a σ-structure, let L be a set of labels and let $\neg L = \{\neg l : l \in L\}$ be another (disjoint) set of labels. A *labelled σ-structure* $\lambda(\mathcal{A})$ is the σ-structure $\mathcal{A}$ with some of its instantiations labelled with some label from the set $L \cup \neg L$. $\square$

When discussing labelled σ-structures, we use the symbols of the signature σ to refer to the underlying instantiations of the labelled σ-structure. For example, if R is a ternary relation symbol in σ and $\lambda(\mathcal{A})$ is a labelled σ-structure then by '$R(a, b, c)$ holds in $\lambda(\mathcal{A})$' we mean '$R(a, b, c)$ holds in $\mathcal{A}$ and there may or may not be a label associated with this instantiation'.

Definition 2 Let $\lambda(\mathcal{A})$ be some labelled σ-structure (with label set $L \cup \neg L$). The *labelled game on* $\lambda(\mathcal{A})$ is defined as follows. The game is played by amending the labelled σ-structure $\lambda(\mathcal{A})$ and there are two players, Player 1 and Player 2, who make alternate moves with Player 1 moving first.

- A move by Player 1 is as follows. Player 1 chooses some label $l \in L$. Player 1 then removes this label from all the instantiations of $\lambda(\mathcal{A})$ which happen to be labelled with l and also removes any instantiation from $\lambda(\mathcal{A})$ which happens to be labelled with $\neg l$. Hence, after such a move by Player 1: there are no instantiations labelled l or $\neg l$; every instantiation previously labelled l appears in the amended structure without a label; and every instantiation previously labelled $\neg l$ no longer appears (labelled or unlabelled) in the amended structure.

- A move by Player 2 is almost identical to a move by Player 1 except that the roles of L and $\neg L$ are reversed. That is, Player 2 chooses some label $\neg l \in \neg L$. Player 2 then removes this label from all the instantiations of $\lambda(\mathcal{A})$ which happen to be labelled with $\neg l$ and also removes any instantiation from $\lambda(\mathcal{A})$ which happens to be labelled with l. Hence, after such a move by Player 2: there are no instantiations labelled l or $\neg l$; every instantiation previously labelled $\neg l$ appears in the amended structure without a label; and every instantiation previously labelled l no longer appears (labelled or unlabelled) in the amended structure.

Players 1 and 2 make moves until there are no more labels left to choose. Once $l \in L$ or $\neg l \in \neg L$ has been chosen by some player, henceforth neither l nor $\neg l$ is ever chosen by any player again. Note that a player might choose a label $l \in L$ or $\neg l \in \neg L$ and there might be no instantiations labelled l or $\neg l$: then $\lambda(\mathcal{A})$ suffers no amendment. What is left at the end of a play is a σ-structure.

The conditions for a play to be a winning play for some player depend on the context. Let Ω be some problem over σ. Player 1 wins a play of the labelled game on $\lambda(\mathcal{A})$ *in the context of* Ω if, and only if, the resulting σ-structure obtained through the moves of Players 1 and 2 is in Ω. □

Definition 3 Let Ω be some problem over σ. An instance of the problem $\mathcal{LG}(\Omega)$ is a labelled σ-structure $\lambda(\mathcal{A})$, and $\lambda(\mathcal{A})$ is a yes-instance if Player 1 has a winning strategy in the labelled game on $\lambda(\mathcal{A})$ in the context Ω. □

Although the sets of labels used in different instances of $\mathcal{LG}(\Omega)$, where Ω is a problem over σ, might be different, we always assume that the size of a label set associated with any σ-structure of size n is bounded by some (fixed) polynomial in n.

Example 4 Consider the labelled σ_{2++}-structure $\lambda(\mathcal{A})$ in Fig. 1. One can show that this is a yes-instance of the problem $\mathcal{LG}(\mathrm{TC})$, i.e., that Player 1 has a winning strategy for the labelled game on $\lambda(\mathcal{A})$ in the context of TC, as follows. Player 1 chooses l and Player 2 must choose $\neg r$ as otherwise Player 1 will choose r so as to yield a path from the source to the sink. Player 1 now chooses q and Player 2 can not choose $\neg p$ as this would yield a path from the source to the sink. Player 1 now chooses p. □

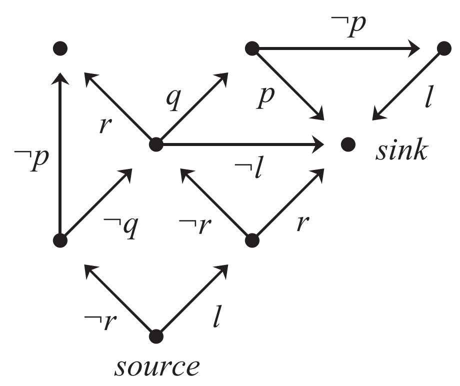

Figure 1. A labelled digraph.

4 Our main result

Lemma 5 *If Ω is a problem in* **PSPACE** *then $\mathcal{LG}(\Omega)$ is in* **PSPACE***.*

Proof (*Sketch*) Suppose that Ω is a problem over σ. In order to verify whether Player 1 has a winning strategy in the labelled game on some labelled σ-structure in the context of Ω, we proceed as follows. Observe that we can store a play of the game as a list of label choices made by the players. We begin by guessing a choice for Player 1 and storing this as the first element of our list (note that it suffices to exhibit a non-deterministic polynomial-space algorithm as **PSPACE** = **NPSPACE**). For every possible choice for Player 2, which we make in turn and record as the second element of the list, we guess a choice for Player 1, which we record as the third element of the list, and we continue in this fashion. As Ω is in **PSPACE**, we can evaluate using polynomial-space whether the resulting σ-structure obtained from a play is in Ω. The space used in our 'house-keeping' (as regards ensuring that we check, for example, that every possible first choice for Player 2 has actually been checked) is again polynomial and so the result follows. $\square$

The following problem will be central to what follows. The problem HEX-EDGES is a problem over the signature σ_{2++} and consists of those digraphs G (equipped with a source s and a sink t) for which Player 1 has a winning strategy in the following 2-player game called *Hex-Edges*. A play of the game consists of Player 1 making the first move and the 2 players moving alternately thereafter until there are no unmarked edges remaining. Initially, all edges of G are unmarked. A move by Player 1 consists of marking an unmarked edge of G as 'retained'; and a move of Player 2 consists of marking an unmarked edge of G as 'dispensed'. Player 1 wins a play of the game if, and only if, when all edges have been marked there is a path of edges marked 'retained' leading from the source s to the sink t. The problem HEX-EDGES was proven to be **PSPACE**-complete by Even and Tarjan [2].

We are now in a position to prove our main result.

Theorem 6 *Suppose that there is a first-order projection with successor from TC to some problem Ω. Then there is a log-space reduction from HEX-EDGES to $\mathcal{LG}(\Omega)$. Hence, $\mathcal{LG}(\Omega)$ is complete for* **PSPACE** *via log-space reductions.*

Proof Let ρ be our first-order projection with successor from TC to Ω, and let us suppose that Ω is over σ. Consider some σ_{2++}-structure $\mathcal{A}$ of size n. From our earlier discussion of first-order projections with successor, for any relation symbol R (of arity a) of σ and for any tuple $\mathbf{u}$ (of length a) over $|\mathcal{A}|$, as to whether $R(\mathbf{u})$ holds in $\rho(\mathcal{A})$ depends upon at most one instantiation of $\mathcal{A}$. There are four possibilities.

(*a*) $R(\mathbf{u})$ holds in $\rho(\mathcal{A})$ if, and only if, $E(a,b)$ holds in $\mathcal{A}$, for some $a, b \in |\mathcal{A}|$.

(*b*) $R(\mathbf{u})$ holds in $\rho(\mathcal{A})$ if, and only if, $\neg E(a,b)$ holds in $\mathcal{A}$, for some $a, b \in |\mathcal{A}|$.

(*c*) For every σ_{2++}-structure $\mathcal{B}$ of size n, $R(\mathbf{u})$ holds in $\rho(\mathcal{B})$.

(*d*) For every σ_{2++}-structure $\mathcal{B}$ of size n, $\neg R(\mathbf{u})$ holds in $\rho(\mathcal{B})$.

Let us construct a labelled σ-structure $\lambda(\rho(\mathcal{A}))$ according to the cases above. For every relation symbol R (of arity a) of σ and for every tuple $\mathbf{u}$ (of length a) over $|\rho(\mathcal{A})|$:

- if case (a) applies and $E(a,b)$ holds in $\mathcal{A}$ then $R(\mathbf{u})$ holds in $\lambda(\rho(\mathcal{A}))$ and we label this instantiation with the label $E(a,b)$;
- if case (a) applies and $\neg E(a,b)$ holds in $\mathcal{A}$ then $R(\mathbf{u})$ does not hold in $\lambda(\rho(\mathcal{A}))$;
- if case (b) applies and $E(a,b)$ holds in $\mathcal{A}$ then $R(\mathbf{u})$ holds in $\lambda(\rho(\mathcal{A}))$ and we label this instantiation with the label $\neg E(a,b)$;
- if case (b) applies and $\neg E(a,b)$ holds in $\mathcal{A}$ then $R(\mathbf{u})$ holds in $\lambda(\rho(\mathcal{A}))$ and we do not attach any label to this instantiation;
- if case (c) applies then $R(\mathbf{u})$ holds in $\lambda(\rho(\mathcal{A}))$ and we do not attach any label to this instantiation; and
- if case (d) applies then $R(\mathbf{u})$ does not hold in $\lambda(\rho(\mathcal{A}))$.

Note that the labels actually appearing in $\lambda(\rho(\mathcal{A}))$ come from $\{E(a,b), \neg E(a,b) : a,b \in |\mathcal{A}|, E(a,b) \text{ holds in } \mathcal{A}\}$; that is, the edges of the digraph $\mathcal{A}$ (recall that when a constant is defined via a first-order projection with successor, the defining formula contains no symbols from the underlying signature and so a constant in $\lambda(\rho(\mathcal{A}))$ depends solely on the size of $\mathcal{A}$). We define the label set L to be $\{E(a,b) : a,b \in |\mathcal{A}|, E(a,b) \text{ holds in } \mathcal{A}\}$ and the label set $\neg L$ to be $\{\neg E(a,b) : a,b \in |\mathcal{A}|, E(a,b) \text{ holds in } \mathcal{A}\}$.

We associate moves by the players in the game of Hex-Edges on $\mathcal{A}$ and moves by the players in the labelled game on $\lambda(\rho(\mathcal{A}))$ as follows.

- If Player 1 in the game of Hex-Edges on $\mathcal{A}$ chooses to include an edge $E(a,b)$ of the digraph $\mathcal{A}$ then Player 1 in the labelled game on $\lambda(\rho(\mathcal{A}))$ chooses the label $E(a,b)$.
- If Player 2 in the game of Hex-Edges on $\mathcal{A}$ chooses to exclude an edge $E(a,b)$ of the digraph $\mathcal{A}$ then Player 2 in the labelled game on $\lambda(\rho(\mathcal{A}))$ chooses the label $\neg E(a,b)$.
- If Player 1 in the labelled game on $\lambda(\rho(\mathcal{A}))$ chooses the label $E(a,b)$ then Player 1 in the game of Hex-Edges on $\mathcal{A}$ chooses to include the edge $E(a,b)$ of the digraph $\mathcal{A}$.
- If Player 2 in the labelled game on $\lambda(\rho(\mathcal{A}))$ chooses the label $\neg E(a,b)$ then Player 2 in the game of Hex-Edges on $\mathcal{A}$ chooses to exclude the edge $E(a,b)$ of the digraph $\mathcal{A}$.

Suppose that Player 1 has a winning strategy in the game of Hex-Edges on $\mathcal{A}$ and suppose that Player 1's first move in this winning strategy is α. Then in the labelled game on $\lambda(\rho(\mathcal{A}))$, Player 1's first move is the dual move $\delta(\alpha)$ of α (dual according to the association described above). Suppose that Player 2 replies in the labelled game on $\lambda(\rho(\mathcal{A}))$ with the move β. Make the dual move $\delta(\beta)$ by Player 2 in the game of Hex-Edges on $\mathcal{A}$ and suppose that Player 1 replies in the game of Hex-Edges on $\mathcal{A}$

with the move γ. Then in the labelled game on $\lambda(\rho(\mathcal{A}))$ make the dual move $\delta(\gamma)$ by Player 1. Thus, by continuing in this fashion, we obtain a strategy for Player 1 in the labelled game on $\lambda(\rho(\mathcal{A}))$. However, note how the two remaining structures, i.e., the σ_{2++}-structure $\mathcal{A}'$ remaining after the play of the game of Hex-edges on $\mathcal{A}$ and the σ-structure $\rho(\mathcal{A})'$ remaining after the play of the labelled game on $\lambda(\mathcal{A})$, are such that $\rho(\mathcal{A})'$ is none other than $\rho(\mathcal{A}')$. Hence, $\mathcal{A}'$ is a yes-instance of TC if, and only if, $\rho(\mathcal{A})'$ is a yes-instance of Ω, i.e., Player 1 wins the game of Hex-Edges on $\mathcal{A}$ if, and only if, Player 1 wins the labelled game on $\lambda(\rho(\mathcal{A}))$ in the context of Ω. Thus, if Player 1 has a winning strategy in the game of Hex-Edges on $\mathcal{A}$ then Player 1 has a winning strategy in the labelled game on $\lambda(\rho(\mathcal{A}))$ in the context of Ω. The converse is obtained similarly and the result follows from Lemma 5 and [2] (as the labelled σ-structure $\lambda(\rho(\mathcal{A}))$ can clearly be constructed from $\mathcal{A}$ using $O(\log n)$ space). $\square$

Note in the proof of Theorem 6 our reliance on the reduction from TC to Ω being a (first-order) projection with successor. Our target structure $\lambda(\rho(\mathcal{A}))$ will not be labelled with one 'bit' of the source structure unless our reduction is a projection. Note also the generality of (the proof of) Theorem 6. Not only can we take the problem Ω to be *any* problem complete via first-order projections with successor for some complexity class containing **NL** (there are many such problems: see, for example, [4, 6, 7]): the proof also holds in the situation where we have a first-order projection with successor from *any* problem Ω' to any problem Ω. We focus on TC because the resulting game, that is, Hex-Edges, has been previously studied and its complexity classified; and we use this classification as our starting point.

As an illustration of how we might apply Theorem 6, by [4] there is a first-order projection with successor ρ such that:

- for every σ_{2++}-structure $\mathcal{A}$, $\rho(\mathcal{A})$ is a $\sigma_{2,2}$-structure such that when considered as a Boolean formula in conjunctive normal form, every clause has exactly 0 or 2 literals; and
- $\mathcal{A} \in$ TC if, and only if, $\rho(\mathcal{A})$ is satisfiable.

Applying the proof of Theorem 6 to the first-order projection with successor ρ, we obtain that the following decision problem LABELLED-4-SAT is complete for **PSPACE** via log-space reductions.

- An instance is a Boolean formula $\mathcal{C}$ in conjunctive normal form where every clause has exactly 4 literals, the *ground literals*, over some set of Boolean variables G, and where associated with each occurrence of a literal in some clause is another literal, a *label literal*, over a set L of Boolean variables disjoint from G such that the label literals associated with some clause are X, $\neg X$, Y and $\neg Y$, for some (possibly identical) Boolean variables X and Y from L.
- A yes-instance is an instance where Player 1 has a winning strategy in the following game:
  - Players 1 and 2 alternate in making moves, with Player 1 moving first, where a move by Player 1 consists of choosing a positive label literal and a move by Player 2 consists of choosing a negative label literal (where once a label literal is chosen neither it nor its negation can be chosen by any player henceforth);

- a play ends when for every Boolean variable Z from L, either Z or $\neg Z$ has been chosen by some player; and
- Player 1 wins a play if, and only if, the 2-CNF Boolean formula obtained from $\mathcal{C}$ by only including those occurrences of ground literals in clauses where the associated label literal has been chosen by either Player 1 or Player 2 is satisfiable.

Example 7 Consider the labelled conjunctive normal form formula in Fig. 2. If Player 1 begins by choosing the label literal Z then no matter how Player 2 plays, Player 1 can choose a positive label literal so that the resulting conjunctive normal form formula (obtained at the end of the play) has exactly two ground literals in each of its clauses and is satisfiable. Hence, the instance in Fig. 2 is a yes-instance of LABELLED-4-SAT. □

$$(\overset{Z}{c} \vee \overset{\neg Z}{a} \vee \overset{X}{a} \vee \overset{\neg X}{a}) \wedge (\overset{X}{\neg b} \vee \overset{\neg X}{c} \vee \overset{Y}{\neg b} \vee \overset{\neg Y}{b}) \wedge (\overset{Z}{\neg c} \vee \overset{\neg Z}{\neg a} \vee \overset{Y}{b} \vee \overset{\neg Y}{\neg a})$$

Figure 2. An instance of LABELLED-4-SAT.

The problem LABELLED-4-SAT can be further transformed into a problem whose instances involve only (unlabelled) Boolean formulae; for we can transform the label literals into ground literals. Consider some instance I of LABELLED-4-SAT in which some ground literal γ, say, is labelled with the label literal Λ. We can replace this occurrence in the clause with the formula $(\Lambda \wedge \gamma)$. Hence, the instance I can be thought of as a conjunction of disjunctions of 4 conjunctions involving 2 literals where the two literals come from disjoint sets, the first being a label literal and the second being a ground literal. Furthermore, the label literals of each pair in a disjunction of 4 conjunctions are always of the form X, Y, $\neg X$ and $\neg Y$, for some Boolean variables X and Y from G. Call this new instance $f(I)$. It is straight-forward to see that I is a yes-instance of LABELLED-4-SAT if, and only if, Player 1 has a winning strategy in the following two-player game:

- Players 1 and 2 alternate in making moves, with Player 1 moving first, where a move by Player 1 consists of setting a positive label literal to true and a move by Player 2 consists of setting a negative label literal to true;
- a play ends when every label literal has been set as either true or false; and
- Player 1 wins a play if, and only if, the 2-CNF Boolean formula remaining is satisfiable (note that the remaining CNF Boolean formula is indeed a 2-CNF Boolean formula and can be evaluated in polynomial-time).

As further applications of Theorem 6, consider the following two problems. The problem 2AD2 consists of those σ_{2++}-structures which, when realised as digraphs: are acyclic; are such that every vertex has in- and out-degree at most 2; and have a path from vertex C to vertex D. Define $\sigma_2 = \langle E \rangle$, where E is a binary relation symbol. The problem STRONG consists of those σ_2-structures which, when realised as digraphs, are strongly connected. Both 2AD2 and STRONG were proven in [7] to be complete for **NL** via first-order projections with successor. Moreover, it was shown that any problem Ω in **NL** can be reduced to 2AD2 by a first-order projection with successor so that *any* instance of Ω (yes or no) is reduced to an acyclic digraph for which every vertex has in- and out-degree at most 2. Theorem 6 implies that the problems $\mathcal{LG}(\text{2AD2})$ and $\mathcal{LG}(\text{STRONG})$ are **PSPACE**-complete via log-space reductions.

5 Some additional remarks

We have seen that we can use the existence of a first-order projection with successor from TC to a problem Ω to automatically show that the problem $\mathcal{LG}(\Omega)$ is **PSPACE**-complete. Our approach is applicable to many problems Ω. However, using this approach, the computational complexity of Ω will always be **NL**-hard. A question arises as to whether there exist problems Ω of 'lower' complexity, e.g., in **L**, for which the problem $\mathcal{LG}(\Omega)$ is still **PSPACE**-complete. The answer is 'yes'.

Consider the following problem BAGS over the signature $\sigma_{2,2} = \langle P, N\rangle$ (where P and N are relation symbols of arity 2). We interpret a $\sigma_{2,2}$-structure $\mathcal{A}$ as a description of whether certain persons either put a peach or a nectarine in a collection of bags via: if $P(i,j)$ holds in $\mathcal{A}$ then person i puts a peach into bag j; and if $N(i,j)$ holds in $\mathcal{A}$ then person i puts a nectarine in bag j. A $\sigma_{2,2}$-structure $\mathcal{A}$ is in BAGS if, and only if, $\mathcal{A}$ describes a collection of bags with the property that every non-empty bag contains at least one peach. The problem BAGS can be described by the following first-order sentence:

$$\forall y(\exists x(P(x,y) \vee N(x,y)) \Rightarrow \exists x P(x,y)).$$

It is easy to see that there is an encoding of the problem BAGS as a set of strings over $\{0,1\}$ and a deterministic Turing machine M running in constant space such that M accepts exactly those strings encoding yes-instances of BAGS.

We shall now show that $\mathcal{LG}(\text{BAGS})$ is **PSPACE**-complete. In order to do this, we shall use the fact that the following problem VPTA (standing for Variable-Partition-Truth-Assignment) is **PSPACE**-complete. A σ_2-structure $\mathcal{A}$ can be interpreted as a collection of clauses of Boolean variables via: $E(i,j)$ holds in $\mathcal{A}$ if, and only if, Boolean variable X_i appears in clause C_j. The problem VPTA consists of those σ_2-instances for which Player 1 has a winning strategy in the following two-player game:

- Players 1 and 2 alternate in making moves, with Player 1 moving first, where a move by Player 1 consists of setting a Boolean variable to true and a move by Player 2 consists of setting a Boolean variable to false;
- a play ends when every Boolean variable has been set as either true or false; and
- Player 1 wins a play if, and only if, the resulting truth assignment is a satisfying truth assignment of the original collection of clauses.

The problem VPTA was proven to be **PSPACE**-complete by Shaefer [5].

We shall now show that there is a log-space reduction from VPTA to $\mathcal{LG}(\text{BAGS})$. Consider an instance I of VPTA. If the Boolean variable X_i is in clause C_j then we shall insist that person i place a peach in bag j and we shall label this peach with the Boolean variable X_i, and we shall also insist that person i places a nectarine in bag j and we shall label this nectarine with the Boolean literal $\neg X_i$. This results in an instance $f(I)$ of $\mathcal{LG}$(BAGS). Note that the non-empty bags of $f(I)$ correspond in a one-to-one way with the non-empty clauses in I; and that no matter what the play in the game $\mathcal{LG}$(BAGS) on $f(I)$, the resulting non-empty bags will be exactly the non-empty bags at the beginning. It is trivial to see that I is a yes-instance of VPTA if, and only if, $f(I)$ is a yes-instance of $\mathcal{LG}$(BAGS). Hence, we obtain the following.

Proposition 8 *$\mathcal{LG}$(BAGS) is* **PSPACE***-complete.* □

We conclude that there does not appear to be any obvious link between the complexity of a problem Ω and the complexity of the resulting game-problem $\mathcal{LG}(\Omega)$.

The reader may be attempting to apply the following line of reasoning. "If there is a first-order projection with successor from BAGS to some problem Ω then there must be a log-space reduction from $\mathcal{LG}$(BAGS) to $\mathcal{LG}(\Omega)$. As $\mathcal{LG}$(BAGS) is **PSPACE**-complete then so $\mathcal{LG}(\Omega)$ is **PSPACE**-complete. As there is a first-order projection with successor from BAGS to TC and first-order projections with successor are transitive then Theorem 6 should be phrased around BAGS instead of TC (for more applicability)." However, the very first assumption in this line of reasoning fails (or at least it does not follow from the construction in the proof of Theorem 6). Note that in the proof of Theorem 6, the game we play on an instance of TC, that is, Hex-Edges, is not the same as the game we play on a labelled digraph: what we exploited was that the game of Hex-Edges had been previously classified, in a complexity-theoretic sense. Nevertheless, it still may be the case that the existence of a first-order projection with successor from BAGS to some problem Ω implies that $\mathcal{LG}(\Omega)$ is **PSPACE**-complete. We do not pursue this point any further in this note as our main intention has been to establish a link between restricted logical reductions and automatic proofs of completeness.

References

[1] H.-D. Ebbinghaus and J.Flum, *Finite Model Theory*, Springer Verlag (1995).

[2] S. Even and R.E. Tarjan, A combinatorial problem which is complete in polynomial space, *J. Assoc. Comput. Mach.* **23** (1976) 710–719.

[3] N. Immerman, Languages that capture complexity classes, *SIAM J. Comput.* **16** (1987) 760–778.

[4] N. Immerman, *Descriptive Complexity*, Springer-Verlag (1998).

[5] T.J. Schaefer, Complexity of some two-person perfect-information games, *J. Comput. System Sci.* **16** (1978) 185–225.

[6] I.A. Stewart, Methods for proving completeness via logical reductions, *Theoret. Comput. Sci.* **118** (1993) 193–229.

[7] I.A. Stewart, Completeness for path-problems via logical reductions, *Inform. Computat.* **121** (1995) 123–134.