Computing Laboratory

Complete problems involving Boolean labelled structures and projection translations

I.A. Stewart

Technical Report Series
No 335 August, 1991
Complete problems involving Boolean labelled structures and projection translations

I. A. Stewart

Abstract

We show that the classes of the Polynomial Hierarchy have complete problems via projection translations and we exhibit some new, natural complete problems for the complexity classes \( \text{co-NP} \) and \( \Pi_2^P \): these problems were not even known to be complete for \( \text{co-NP} \) and \( \Pi_2^P \) via polynomial-time reductions. Most of the problems involve the reliability of networks of processors where the links may fail and where these failures may be related to one another in a restricted way, but we show how our trick of labelling discrete structures with Boolean literals can be applied to yield other new natural complete problems involving digraphs and Boolean formulae. We also show that it is unlikely that certain problems (known to be complete for the respective complexity class via projection translations) are complete for \( L \), \( NL \), and \( NP \) via monotone projection translations; for if any of them are then \( L = NP \), \( NL = NP \), or \( NP = \text{co-NP} \).
Bibliographical details

STEWARD, Iain Alastair

Complete problems involving Boolean labelled structures and projection translations.
[By] I. A. Stewart.

Newcastle upon Tyne: University of Newcastle upon Tyne: Computing Laboratory, 1991.

(University of Newcastle upon Tyne, Computing Laboratory, Technical Report Series, no. 335)

Added entries

UNIVERSITY OF NEWCASTLE UPON TYNE.
Computing Laboratory. Technical Report Series. 335

Abstract

We show that the classes of the Polynomial Hierarchy have complete problems via projection translations and we exhibit some new, natural complete problems for the complexity classes co-NP and $\Pi_2^P$; these problems were not even known to be complete for co-NP and $\Pi_2^P$ via polynomial-time reductions. Most of the problems involve the reliability of networks of processors where the links may fail and where these failures may be related to one another in a restricted way, but we show how our trick of labelling discrete structures with Boolean literals can be applied to yield other new natural complete problems involving digraphs and Boolean formulae. We also show that it is unlikely that certain problems (known to be complete for the respective complexity class via projection translations) are complete for $L$, $NL$, and $NP$ via monotone projection translations; for if any of them are then $L = NP$, $NL = NP$, or $NP = co-NP$.

About the author

Dr. Stewart has been a lecturer at the Computing Laboratory of the University of Newcastle upon Tyne since 1986.

Suggested keywords

COMPLETE PROBLEMS
DESCRIPTIVE COMPLEXITY THEORY
FINITE MODEL THEORY
NETWORK RELIABILITY
POLYNOMIAL HIERARCHY

Suggested classmarks (primary classmark underlined)

Dewey (18th): 511.3 511.8
U.D.C. 510.52 510.64
Complete Problems involving Boolean Labelled Structures and Projection Translations

Iain A. Stewart
Computing Laboratory, Univ. of Newcastle upon Tyne, Claremont Tower,
Claremont Road, Newcastle upon Tyne, NE1 7RU, England.

1. Introduction.

We are concerned in this paper with the existence of complete problems for various complexity classes via projection translations (an extremely weak form of reduction). These logical translations originated in [Imm87] where L, NL, and P were shown to have such complete problems. Also, in [Ste91a], [Ste91b], and [Ste91c], various problems were shown to be complete for NP via projection translations (resp. without successor). All complete problems involved were already known to be complete for the various complexity classes via logspace reductions.

Projection translations are of interest for various reasons. If a problem \( \Omega \) is complete for NP, say, via polynomial-time reductions and we can show that \( \Omega \) is in P then \( P = NP \): if we can show that \( \Omega \) is in L then we can make no stronger deduction unless we know that \( \Omega \) is complete for NP via logspace reductions also. Consequently, the weakness of the projection translation enables us to make even stronger statements concerning problems complete for NP via projection translations and complexity classes within L. Also, as Immerman pointed out [Imm87], it seems plausible, due to the fact that projection translations are very restrictive, to show that projection translations (of a certain arity) do not exist between certain problems. This suggestion should be compared with the approach taken by Ajtai and Fagin [AF90] who showed that certain problems cannot be expressed in certain logics. It is hoped that such (in)expressibility results might lead to a deeper understanding of what makes a problem hard to compute on a model of computation such as the resource-bounded Turing machine. (We remark that throughout this paper we focus on projection translations as opposed to projection translations without successor, because the successor relation appears
to be required to capture complexity classes within \( \text{NP} \) and because it occurs naturally when representing data in computers.)

In this paper, we show that the classes \( \{\Sigma_k^p, \Pi_k^p : k = 0,1,2,...\} \) of the Polynomial Hierarchy also have complete problems via projection translations (without successor) and we exhibit some new, natural complete problems for the complexity classes \( \text{co-NP} \) and \( \Pi_2^p \); these problems were not even known to be complete for \( \text{co-NP} \) and \( \Pi_2^p \) via polynomial-time reductions. Most of the problems involve the reliability of networks of processors where the links may fail and where these failures may be related to one another in a restricted way: the consideration of reliability in such networks where the failures may be arbitrarily related is extremely difficult ([Joh85]).

Our techniques involve the characterization of complexity classes using logic, and, in particular, use the fact that these logics have normal forms involving projective formulae (so enabling us to prove completeness results). The proofs of the results obtained in this paper depend crucially on the logical characterization of complexity classes, and it is hard to see how the results might have been proven otherwise. Our methods and results lend further credance to the view that projection translations should be intensively studied in their own right. We also use our techniques to show that it is probably the case that the complexity classes \( \mathcal{L}, \mathcal{NL}, \) and \( \text{NP} \) do not have complete problems via monotone projection translations: these reductions are even weaker than projection translations.

This Introduction is §1 and the basic definitions and related results are given in §2. In §3 we show that the classes \( \{\Sigma_k^p, \Pi_k^p : k = 0,1,2,...\} \) of the Polynomial Hierarchy have complete problems via projection translations: essentially, we generalize (but refine) techniques used by Dahlhaus ([Dah84]) to show that various problems are complete for \( \text{NP} \) via interpretative reductions (another logical reduction, though provably different from projection translations [Ste91c]). These complete problems were already known to be complete for the various classes via logspace reductions. In §4 we exhibit some new complete problems for \( \text{co-NP} \) and \( \Pi_2^p \) (via projection translations). Although these problems involve reliability in networks, we show how our trick of labelling discrete structures with Boolean literals can be applied to yield other complete problems involving digraphs and Boolean formulae. We also point out that for our techniques to work, we require that a problem be complete for some (smaller) complexity class via projection translations as opposed to other more general reductions, so emphasising the importance of the projection translation as a reduction between problems. In §5 we use the same techniques of §3 and §4 to show that it is unlikely that certain problems (known to be complete for the respective complexity class
via projection translations) are complete for L, NL, and NP via monotone projection translations; for if any of them are then L = NP, NL = NP, or NP = co-NP. Our conclusions are given in §6. We add that throughout complexity classes and tuples are written in bold type.

2. Basic definitions and related results.

In this section we describe how we extend first-order logic using an operator corresponding to some decision problem and we consider the notion of a logical translation between problems. We also mention some relevant existing results. The reader is referred to [End72], [Imm87], [Ste91a], [Ste91b], and [Ste91c] for extensive details of the concepts mentioned here.

A vocabulary \( \tau = \langle R_1, R_2, \ldots, R_k, C_1, C_2, \ldots, C_m \rangle \) is a tuple of relation symbols \( \{ R_i : i = 1, 2, \ldots, k \} \), with \( R_i \) of arity \( a_i \), and constant symbols \( \{ C_i : i = 1, 2, \ldots, m \} \). A (finite) structure of size \( n \) over \( \tau \) is a tuple \( S = \langle \{0,1,\ldots,n-1\}, R_1, R_2, \ldots, R_k, C_1, C_2, \ldots, C_m \rangle \) consisting of a universe \(| S | = \{0,1,\ldots,n-1\} \), relations \( R_1, R_2, \ldots, R_k \) on the universe \(| S |\) of arities \( a_1, a_2, \ldots, a_k \), respectively, and constants \( C_1, C_2, \ldots, C_m \) from the universe \(| S |\). The size of some structure \( S \) is also denoted by \(| S |\). We denote the set of all structures over \( \tau \) by \( \text{STRUCT}(\tau) \) (henceforth, we do not distinguish between relations (resp. constants), and relation (resp. constant) symbols, and we assume that all structures are of size at least 2). If \( \tau' \) is a vocabulary containing all the symbols of the vocabulary \( \tau \) and \( S' \in \text{STRUCT}(\tau') \), then \( S|_{\tau} \) is the structure over \( \tau \) obtained by restricting \( S' \) to the symbols of \( \tau \) only.

A problem of arity \( t \) (\( \geq 0 \)) over \( \tau \) is a subset of \( \text{STRUCT}_t(\tau) = \{ (S,u) : S \in \text{STRUCT}(\tau); u \in |S|^t \} \) (we remark that we usually refer to a set of strings as a decision problem as opposed to a problem, which we reserve for a set of finite structures). If \( \Omega \) is some problem then \( \tau(\Omega) \) denotes its vocabulary.

The language of the first-order logic \( FO_{\leq}(\tau) \) has as its (well-formed) formulae those formulae built, in the usual way, from the relation and constant symbols of \( \tau \), the binary relation symbols \( = \) and \( s \), and the constant symbols \( 0 \) and \( \max \), using the logical connectives \( \lor \), \( \land \), and \( \neg \), the variables \( \{ x, y, z_3, \ldots \} \), and the quantifiers \( \exists \) and \( \forall \). Any formula \( \phi \) of \( FO_{\leq}(\tau) \), with free variables those of the \( t \)-tuple \( x \), is interpreted in the set \( \text{STRUCT}_t(\tau) \), and for each \( S \in \text{STRUCT}(\tau) \) of size \( n \) and \( u \in |S|^t \), we have that \((S,u) \models \phi(x) \) if and only if \( \phi^S(u) \) holds, where \( \phi^S(u) \) denotes the obvious interpretation of \( \phi \) in \( S \), except that the binary relation symbol \( = \) is always interpreted in \( S \) as equality, the binary relation symbol \( s \) is interpreted as the successor relation on \(| S |\), the constant symbol \( 0 \) is interpreted as \( 0 \in |S| \), the constant symbol \( \max \) is interpreted as \( n-1 \in |S| \), and each variable of \( x \)
is given the corresponding value from u. (We usually write \( s(x,y) \) as \( y = x + 1 \), and \( \neg(x = y) \) as \( x \neq y \).) If we forbid the use of the successor relation in the logic \( \text{FO}_\leq(\tau) \) then we denote the resulting logic by \( \text{FO}(\tau) \): also, \( \text{FO}_\leq = \bigcup \{ \text{FO}_\leq(\tau) : \tau \text{ some vocabulary} \} \) (with \( \text{FO} \) defined similarly). The formula \( \phi \) describes (or specifies or represents) the problem:

\[
\{ (S, u) : (S, u) \in \text{STRUCT}_1(\tau), (S, u) \models \phi(x) \} \text{ of arity } t.
\]

Having detailed how we use first-order logic to describe problems, we now illustrate how we extend first-order logic with new operators to attain greater expressibility. Let \( \tau_2 \) be the vocabulary consisting of the binary relation symbol \( E \): so, we may clearly consider structures \( S \) over \( \tau_2 \) as digraphs or graphs (for \( i \neq j \), there is an edge \( (i,j) \) in the digraph \( S \) if and only if \( E_S(i,j) \) holds, and there is an edge \( (i,j) \) in the graph \( S \) if and only if \( E_S(i,j) \) or \( E_S(j,i) \) holds: we assume throughout that there are no edges from a vertex to itself in any graph or digraph). Consider the problem DTC of arity 2:

\[
\text{DTC} = \{ (S, u, v) \in \text{STRUCT}_2(\tau_2) : \text{there is a path in the digraph } S \text{ from vertex } u \text{ to vertex } v \text{ such that each vertex on the path, except perhaps } v, \text{ has out-degree 1, i.e. the path is deterministic}. \}
\]

We write \((\pm \text{DTC})^*[\text{FO}_\leq]\) to denote the logic formed by allowing an unlimited number of nested applications of the operator DTC, where \( \text{DTC}[\lambda xy \psi_S(x,y)] \), for some formula \( \psi \in (\pm \text{DTC})^*[\text{FO}_\leq] \), some \( k \)-tuples of distinct variables \( x \) and \( y \), and some relevant structure \( S \), denotes the digraph with vertices indexed by the tuples of \( |S|^k \), and where there is an edge from \( u \) to \( v \) if and only if there is a deterministic path in the digraph described by \( \psi_S(x,y) \) from \( u \) to \( v \) (this is the logic \( (\text{FO} + \text{DTC}) \) of \( \text{Imm} \text{77} \)). We write \((\pm \text{DTC})^k[\text{FO}_\leq] \) (resp. \( \text{DTC}^k[\text{FO}_\leq] \)) to denote the sub-logic of \((\pm \text{DTC})^*[\text{FO}_\leq] \) where all formulae have at most \( k \) nested applications of the operator DTC (resp. where no operator appears within a negation sign): the sub-logic, \( \text{DTC}^k[\text{FO}_\leq] \), of \( \text{DTC}^*[\text{FO}_\leq] \) is defined similarly. Needless to say, first-order logic can be extended by other operators as we shall soon see.

In order to compare logical descriptions of decision problems, we use the notion of a logical translation (these translations play an analogous role to logspace and polynomial-time reductions between sets of strings). Let \( \tau' = < R_1, R_2, \ldots, R_k, C_1, C_2, \ldots, C_m > \) be some vocabulary, where each \( R_i \) is a relation symbol of arity \( a_i \) and each \( C_j \) is a constant symbol, and let \( L(\tau) \) be some logic over some vocabulary \( \tau \). Then the formulae of \( \Sigma = \{ \phi_i(x_i), \psi_j(y_j) : i = 1, 2, \ldots, k; j = 1, 2, \ldots, m \} \subseteq L(\tau) \), where:

(i) each formula \( \phi_i \) (resp. \( \psi_j \)) is over the qa\( a_i \) (resp. qa\( q \)) distinct variables \( x_i \) (resp. \( y_j \)), for some positive integer \( q \);

(ii) for each \( j = 1, 2, \ldots, m \) and for each structure \( S \in \text{STRUCT}(\tau) \):
\[ S = (\exists x_1)(\exists x_2)\ldots(\exists x_q)(\psi_1(x_1, x_2, \ldots, x_q) \land \\
(\forall y_1)(\forall y_2)\ldots(\forall y_q)(\psi_1(y_1, y_2, \ldots, y_q) \iff (x_1 = y_1 \land x_2 = y_2 \land \ldots \land x_q = y_q)) \],
are called \( \tau \)-descriptive. For each \( S \in \text{STRUCT}(\tau) \), the \( \tau \)-translation of \( S \) with respect to \( \varepsilon \) is the structure \( S' \in \text{STRUCT}(\tau') \) with universe \( |S'| \), defined as follows:
for all \( i = 1, 2, \ldots, k \) and for any tuples \( (u_1, u_2, \ldots, u_{a_i}) \subseteq |S'| = |S|^q \):
\[ R^S_i(u_1, u_2, \ldots, u_{a_i}) \text{ holds if and only if } (S, (u_1, u_2, \ldots, u_{a_i})) \models \phi_i(x_i), \]
and, for all \( j = 1, 2, \ldots, m \) and for any tuple \( u \in |S'| = |S|^q \):
\[ C_j^S = u \text{ if and only if } (S, u) \models \psi_j(y_j) \]
(tuples are ordered lexicographically, with \((0, 0, \ldots, 0) < (0, 0, \ldots, 1) < (0, 0, \ldots, 2) < \ldots, \)
and so on). Let \( \Omega \) and \( \Omega' \) be problems over the vocabularies \( \tau \) and \( \tau' \), respectively. Let \( \varepsilon \) be a set of \( \tau \)-descriptive formulae from some logic \( L(\tau) \), and for each \( S \in \text{STRUCT}(\tau) \), let \( \sigma(S) \in \text{STRUCT}(\tau') \) denote the \( \tau \)-translation of \( S \) with respect to \( \varepsilon \). Then \( \Omega' \) is an \( L \)-translation of \( \Omega \) if and only if for each \( S \in \text{STRUCT}(\tau) \), \( S \in \Omega \) if and only if \( \sigma(S) \in \Omega' \).

Let \( \phi \in \text{FO}_\leq(\tau) \), for some vocabulary \( \tau \), be of the form:
\[ \phi = \bigvee \{ x_i \land \beta_i : i \in I \} \]
for some finite index set \( I \), where:
(i) each \( \beta_i \) is a conjunction of the logical atomic relations, \( =, \) and their negations;
(ii) each \( \beta_i \) is atomic or negated atomic;
(iii) if \( i \neq j \), then \( \beta_i \) and \( \beta_j \) are mutually exclusive.
Then \( \phi \) is a projective formula. If the successor relation symbol \( s \) does not appear in \( \phi \) (that is, \( \phi \in \text{FO}(\tau) \)), then \( \phi \) is a projective formula without successor, and if each of the \( \beta_i \) (above) is atomic then \( \phi \) is a monotone projective formula.
Consequently, we clearly have the notions of one problem being a first-order translation, a quantifier-free translation, a projection translation, and a monotone projection translation of another, as well as all of these without successor. If \( \Omega \) and \( \Omega' \) are problems and there exist problems \( \Omega_0, \Omega_1, \ldots, \Omega_i \), for some \( i \geq 0 \), such that \( \Omega = \Omega_0, \Omega_{j+1} \) is a projection translation of \( \Omega_j \), for \( j < i \), and \( \Omega_i = \Omega' \), then \( \Omega' \) is an iterated projection translation of \( \Omega \).

As mentioned above, first-order logic has been extended by other operators apart from DTC. Let \( \tau_{2,1} \) be the vocabulary consisting of one relation symbol \( E \) of arity 2 together with one relation symbol \( U \) of arity 1, and let \( \tau_{2,2} \) be the vocabulary consisting of 2 relation symbols \( P \) and \( N \), both of arity 2. We can clearly think of a structure \( S \) over \( \tau_{2,1} \) as an alternating digraph ([Imm87]: the relation \( U^S(i) \) holds if and only if vertex \( i \) is a universal vertex). Also, a structure \( S \) of size \( n \) over \( \tau_{2,2} \) can be considered as a Boolean formula in conjunctive (resp. disjunctive) normal form, c.n.f. (resp. d.n.f.), involving the Boolean variables \( \{X_0, X_1, \ldots, X_{n-1}\} \) and with \( n \) clauses, where the literal \( X_i \) (resp. \( \neg X_i \)) is in the clause.
C_j if and only if P^{S}(i,j) (resp. N^{S}(i,j)) holds. Operators corresponding to the following problems have been previously considered ([Imm87], [Ste91a], [Ste91b], [Ste91c]):

\[ \text{STC} = \{(S,u,v) \in \text{STRUCT}_2(\tau_2) : \text{there is a path in the graph } S \text{ from vertex } u \text{ to vertex } v\}; \]

\[ \text{TC} = \{(S,u,v) \in \text{STRUCT}_2(\tau_2) : \text{there is a path in the digraph } S \text{ from vertex } u \text{ to vertex } v\}; \]

\[ \text{ATC} = \{(S,u,v) \in \text{STRUCT}_2(\tau_{2,1}) : \text{there is an alternating path in the alternating digraph } S \text{ from vertex } u \text{ to vertex } v\}; \]

\[ \text{HP} = \{(S,u,v) \in \text{STRUCT}_2(\tau_2) : \text{there is a Hamiltonian path in the digraph } S \text{ from vertex } u \text{ to vertex } v\}; \]

\[ \text{3COL} = \{S \in \text{STRUCT}(\tau_2) : \text{the graph } S \text{ can be 3-coloured}\}; \]

\[ \text{SAT} = \{S \in \text{STRUCT}(\tau_{2,2}) : \text{there is a satisfying truth assignment for the c.n.f. formula } S\}; \]

\[ \leq 3\text{SAT} = \{S \in \text{STRUCT}(\tau_{2,2}) \text{ such that } S \text{ is a c.n.f. formula and has at most } 3 \text{ literals in each clause : there is a satisfying truth assignment for } S\}; \]

\[ 3\text{SAT} = \{S \in \text{STRUCT}(\tau_{2,2}) \text{ such that } S \text{ is a c.n.f. formula and has exactly } 3 \text{ literals in each clause : there is a satisfying truth assignment for } S\}. \]

The problem DTC(0,max) is defined as:

\[ \text{DTC}(0,\text{max}) = \{S \in \text{STRUCT}(\tau_2) : \text{there is a deterministic path in the digraph } S, \text{ of size } n, \text{ between vertex } 0 \text{ and vertex } n-1\}, \]

with the problems STC(0,max), TC(0,max), ATC(0,max), and HP(0,max) defined similarly.

**Theorem 2.1.** ([Imm87], [Imm88])

(a) \( F_{O} \leq \ L = \text{DTC}\uparrow[F_{O} \leq] = (\pm \text{DTC})^*[F_{O} \leq]; \)

(b) \( \text{NSYMLOG} = \text{STC}\uparrow[F_{O} \leq] = \text{STC}^*[F_{O} \leq]; \)

(c) \( \text{NL} = \text{TC}\uparrow[F_{O} \leq] = (\pm \text{TC})^*[F_{O} \leq]; \)

(d) \( \text{P} = \text{ATC}\uparrow[F_{O} \leq] = (\pm \text{ATC})^*[F_{O} \leq]; \)

(e) \( \text{DTC}(0,\text{max}), \text{STC}(0,\text{max}), \text{TC}(0,\text{max}), \text{and ATC}(0,\text{max}) \) are complete for \( L, \)

NSYMLOG, NL, and P, respectively, via projection translations. \( \Box \)

**Theorem 2.2.** ([Dah84], [Ste91a], [Ste91b], [Ste91c])

(a) \( \text{NP} = \text{HP}\downarrow[F_{O} \leq] = \text{HP}^*[F_{O} \leq]; \)

(b) \( 3\text{COL}\downarrow[F_{O} \leq] = \text{SAT}\downarrow[F_{O} \leq] = \leq 3\text{SAT}\downarrow[F_{O} \leq] = 3\text{SAT}\downarrow[F_{O} \leq] = \text{NP}; \)

(c) \( \text{NP} \cup \text{co-NP} \subseteq 3\text{COL}^3[F_{O} \leq]/\cap SAT^3[F_{O} \leq] \cap \leq 3\text{SAT}^3[F_{O} \leq]/\cap 3\text{SAT}^3[F_{O} \leq]; \)
(d) $3\text{COL}^*[\text{FO}_{\leq}] = (\pm 3\text{COL})^*[\text{FO}_{\leq}] = \text{SAT}^*[\text{FO}] = (\pm \text{SAT})^*[\text{FO}_{\leq}] = \leq 3\text{SAT}^*[\text{FO}_{\leq}] = (\pm \leq 3\text{SAT})^*[\text{FO}_{\leq}] = 3\text{SAT}^*[\text{FO}_{\leq}] = (\pm 3\text{SAT})^*[\text{FO}_{\leq}]$;

(e) $\text{SAT}$ is complete for $\text{NP}$ via projection translations without successor, $\text{HP}(0, \text{max})$, $\leq 3\text{SAT}$, and $3\text{COL}$ are complete for $\text{NP}$ via projection translations, and $3\text{SAT}$ is complete for $\text{NP}$ via iterated projection translations. □

It is open as to whether $3\text{SAT}$ is complete for $\text{NP}$ via projection translations, although Dahlhaus has shown that a different encoding (not over the vocabulary $\tau_{2,2}$) of the 3-Satisfiability Problem (where all the clauses in any instance have exactly 3 literals) is not complete for $\text{NP}$ via interpretative reductions ([Dah84]): using the same techniques, we can show that $3\text{COL}$ is not either [Ste91c]). The reader is also referred to [IL89] and [BIS90] for related results concerning complexity classes "below" P (such as $\text{NC}^1$, L, NL, and DET$^*$).


In this section, we consider the complexity classes $\{\Sigma_k^p : k = 0,1,2,\ldots\}$ of the Polynomial Hierarchy ([Sto77]) and we show that there exist problems which are complete for these classes via projection translations: as far as we know, these are the first such problems to be discovered (each problem had previously been known to be complete for the relevant class via logspace reductions). Essentially, we generalize the techniques of Dahlhaus who showed that $\text{SAT}$ is complete for $\text{NP}$ via interpretative reductions ([Dal84]): the fact that $\text{SAT}$ is complete for $\text{NP}$ via projection translations, a result stated in [Ste91c], follows immediately from Theorem 3.1 below.

**Definition 3.1.** Let BF be some Boolean formula over the set of Boolean variables $X$, where $X$ is partitioned as the disjoint union of its subsets $Y_1$, $Y_2$, ..., $Y_k$, for some $k > 0$: we denote this fact by writing $BF(Y_1,Y_2,\ldots,Y_k)$. Then we write:

"$\exists Y_i$..." to mean "there exists some truth assignment on the variables of $Y_i$ such that ...",

and:

"$\forall Y_i$..." to mean "for all truth assignments on the variables of $Y_i$ ...".

Let $\tau_{2,2}(k)$ denote the vocabulary $\tau_{2,2}$ augmented with the relation symbols $M_1$, $M_2$, ..., $M_k$, all of arity 1 (where $M_i$ is used to specify the set of variables $Y_i$ for some Boolean formula $BF(Y_1,Y_2,\ldots,Y_k)$ in c.n.f. or d.n.f.). We define the problems $\text{CNF}_k$ and $\text{DNF}_k$ over $\tau_{2,2}(k)$ as follows:

$\text{CNF}_k = \{BF(Y_1,Y_2,\ldots,Y_k) \text{ is a c.n.f. Boolean formula}\}$
\[ \exists Y_1 \forall Y_2 \ldots \forall Y_k [BF = True] \]

\[ DNF_k = \{ BF(Y_1, Y_2, \ldots, Y_k) \text{ is a d.n.f. Boolean formula} \] 

\[ \exists Y_1 \forall Y_2 \ldots \forall Y_k [BF = True] \]

where the quantifier \( Q_k \) is \( \exists \) (resp. \( \forall \)) if \( k \) is odd (resp. even) (notice that \( CNF_1 \) is simply a reformulation of SAT).

By [MS72], [SM73], and [Wra77] (see [Sto77]), the following (decision) problems are complete for \( \Sigma_k^p \) via logspace reductions:

\[ B_k = \{ BF(Y_1, Y_2, \ldots, Y_k) \text{ is a Boolean formula} \} \exists Y_1 \forall Y_2 \ldots \forall Y_k [BF = True] \]

\[ 3CNF_k = \{ BF(Y_1, Y_2, \ldots, Y_k) \text{ is a c.n.f. Boolean formula with at most 3 literals} \] 

per clause: \( \exists Y_1 \forall Y_2 \ldots \forall Y_k [BF = True] \), for \( k \) odd;

\[ 3DNF_k = \{ BF(Y_1, Y_2, \ldots, Y_k) \text{ is a d.n.f. Boolean formula with at most 3 literals} \] 

per clause: \( \exists Y_1 \forall Y_2 \ldots \forall Y_k [BF = True] \), for \( k \) even (we call both the disjunctions of a c.n.f. formula and the conjunctions of a d.n.f. formula clauses: this is non-standard).

**Lemma 3.1.** Let \( \phi \) be a first-order formula in prenex normal form; that is:

\[ \phi = Q_I x_1 Q_2 x_2 \ldots Q_k x_k \psi, \]

where each quantifier \( Q_i \) is either \( \forall \) or \( \exists \), and \( \psi \) is quantifier-free. Then \( \phi \) is logically equivalent to a second-order formula \( \phi \) of the form:

\[ \phi = \exists R_1 \ldots \exists R_m \forall y_1 \exists z_1 R_1(y_1, z_1) \land \ldots \land \forall y_m \exists z_m R_m(y_m, z_m) \land \forall w_1 \ldots \forall w_p \psi', \]

where each \( R_i \) is a relation symbol of arity \( h_i + 1 \), each \( y_i \) is a tuple of variables of length \( h_i \), and \( \psi \) is quantifier-free (all variables involved are distinct).

**Proof.** We proceed by induction on the number of existential quantifiers in the prefix \( Q_1 x_1 Q_2 x_2 \ldots Q_k x_k \). Suppose that only one existential quantifier occurs as \( Q_i = \exists \). Then \( \phi \) is clearly logically equivalent to:

\[ \exists R[\forall x_1 \ldots \forall x_{i-1} \exists x_i R(x_1, \ldots, x_{i-1}, x_i) \land \forall x_1 \ldots \forall x_{i-1} \forall x_i [R(x_1, \ldots, x_{i-1}, x_i) \Rightarrow \forall x_{i+1} \ldots \forall x_k \psi]], \]

where \( R \) is a relation symbol of arity \( i \). So, \( \phi \) is logically equivalent to:

\[ \exists R[\forall x_1 \ldots \forall x_{i-1} \exists x_i R(x_1, \ldots, x_{i-1}, x_i) \land \forall x_1 \ldots \forall x_k [R(x_1, \ldots, x_{i-1}, x_i) \Rightarrow \psi]], \]

and the base case follows.

Suppose, as our induction hypothesis, that the result holds whenever the prefix \( Q_1 x_1 Q_2 x_2 \ldots Q_k x_k \) contains less than \( q \) existential quantifiers. Let \( Q_i \) be the first existential quantifier in this prefix; so:

\[ Q_1 x_1 Q_2 x_2 \ldots Q_k x_k = \forall x_1 \ldots \forall x_{i-1} \exists x_i Q_{i+1} x_{i+1} \ldots Q_k x_k. \]

Clearly, as above, \( \phi \) is logically equivalent to:

\[ \exists R[\forall x_1 \ldots \forall x_{i-1} \exists x_i R(x_1, \ldots, x_{i-1}, x_i) \land \forall x_1 \ldots \forall x_{i-1} \forall x_i [R(x_1, \ldots, x_{i-1}, x_i) \Rightarrow Q_{i+1} x_{i+1} \ldots Q_k x_k \psi]], \]
where \( R \) is a relation symbol of arity \( i \). So, \( \phi \) is logically equivalent to:
\[
\exists R[\forall x_1...\forall x_{i-1} \exists x_i R(x_1,...,x_{i-1},x_i) \\
\wedge \forall x_1...\forall x_{i-1} \forall x_i Q_i + 1x_i + 1...Q_k x_k[R(x_1,...,x_{i-1},x_i) \Rightarrow \psi]].
\]
Now, by the induction hypothesis applied to:
\[
\forall x_1...\forall x_{i-1} \forall x_i Q_i + 1x_i + 1...Q_k x_k[R(x_1,...,x_{i-1},x_i) \Rightarrow \psi],
\]
we get that \( \phi \) is logically equivalent to:
\[
\exists R[\forall x_1...\forall x_{i-1} \exists x_i R(x_1,...,x_{i-1},x_i) \wedge \exists R[\forall y_1 \exists z_1 R_1(y_1,z_1) \wedge ... \\
\wedge \forall y_m \exists z_m R_m(y_m,z_m) \wedge \forall w_1...\forall w_p \psi]],
\]
where each \( R_i \) is a relation symbol of arity \( h_i + 1 \), each \( y_j \) is a tuple of variables of length \( h_j \), and \( \psi \) is quantifier-free. Hence, \( \phi \) is logically equivalent to:
\[
\exists R\exists R[\forall x_1...\forall x_{i-1} \exists x_i R(x_1,...,x_{i-1},x_i) \wedge \forall y_1 \exists z_1 R_1(y_1,z_1) \wedge ... \\
\wedge \forall y_m \exists z_m R_m(y_m,z_m) \wedge \forall w_1...\forall w_p \psi],
\]
and the result follows. (All variables involved are distinct.) □

**Theorem 3.1.** When \( k \) is odd (resp. even), \( CNF_k \) (resp. \( DNF_k \)) is complete for \( \Sigma^p_k \) via projection translations without successor.

**Proof.** By Theorem 6 of [Fag74] (and the remark before Theorem 3.4 in [Sto77]), \( \Sigma^p_k \) coincides with the class of problems expressible by a sentence \( \phi \) of second-order logic (with successor) of the form:
\[
\exists G_1,...\exists G_{1,n_1} \forall G_{2,1}...\forall G_{2,n_2}...Q_k G_{k,1}...Q_k G_{k,n_k} \phi,
\]
where each \( G_{i,j} \) is a relation symbol of arity \( a_{i,j} \), \( Q_k \) is the quantifier \( \exists \) (resp. \( \forall \)) if \( k \) is odd (resp. even), and \( \phi \) is a first-order sentence: we may clearly assume that \( \phi \) is in prenex normal form. By Lemma 3.1, \( \phi \) is logically equivalent to a sentence of the form:
\[
\exists F_1...\exists F_m[\exists x_1 \exists z_1 F_1(x_1,z_1) \wedge ... \wedge \forall x_m \exists z_m F_m(x_m,z_m) \wedge \forall y_1...\forall y_p \psi],
\]
where each \( F_i \) is a relation symbol of arity \( b_i + 1 \), each \( x_i \) is a tuple of variables of length \( b_i \), and \( \psi \) is quantifier-free in c.n.f. (all variables involved are distinct). Hence, we may assume that \( \phi \) is of the form:
\[
\exists G_{1,1}...\exists G_{1,n_1} \forall G_{2,1}...\forall G_{2,n_2}...Q_k G_{k,1}...Q_k G_{k,n_k} \exists F_1...\exists F_m \\
[\forall x_1 \exists z_1 F_1(x_1,z_1) \wedge ... \wedge \forall x_m \exists z_m F_m(x_m,z_m) \wedge \forall y_1...\forall y_p \psi],
\]
where:
\[
\psi = \psi_1 \wedge \psi_2 \wedge ... \wedge \psi_p,
\]
with each \( \psi_q \) being the disjunction \( \theta_q \lor x_q \), where \( \theta_q \) is the disjunction of all those atomic or negated atomic formulae of \( \psi_q \) involving any of the relation symbols of \( \{G_{i,j},F_j\} \), and \( x_q \) is the disjunction:
\[
xq_1 \lor xq_2 \lor ... \lor xq_r,
\]
of the remaining atomic or negated atomic formulae of \( \psi_q \) (we may assume that each \( \chi_q \) is non-empty). We may also assume that \( n_1 = n_2 = \ldots = n_k = a_{1,1} = a_{1,2} = \ldots = a_{k,n_k} = m = b_1 = b_2 = \ldots = b_m = p = r = r_1 = r_2 = \ldots = r_p \); denote this constant by \( m \).

Let \( \Omega \) be the problem represented by \( \Phi \) where \( k \) is odd, and let \( S \in \text{STRUCT}(\tau(\Omega)) \) be of size \( n \). We are now in a position to translate \( S \) into an instance \( \sigma(S) \) of CNF\(_k\). The instance \( \sigma(S) \) of CNF\(_k\) corresponding to \( S \) consists of the following set of non-empty clauses:

\[
\{B_r : r = 1,2,\ldots,m\} \cup \{C_r(t),D_r(t) : r = 1,2,\ldots,m; t \in |S|^m\},
\]

involving the set of Boolean variables:

\[
\{u_{ij}(t'),v_{ij}(t),w_j : i = 1,2,\ldots,k; j = 1,2,\ldots,m; t \in |S|^m; t' \in |S|^{m+1}\}.
\]

For each tuple \( t \in |S|^m \) and \( r, q = 1, 2, \ldots, m \):

(i) \( B_r = \{w_r\} \);

(ii) \( C_r(t) = \{u_r(t,0),u_r(t,1),\ldots,u_r(t,n-1)\} \);

(iii) \( \neg w_r \in D_r(t) \);

(iv) \( \{w_q, \neg w_q\} \subseteq D_r(t) \iff (S,t) \models \chi_{r,q}(y_1,y_2,\ldots,y_m) \);

(v) consider any occurrence of \( F_j \) or \( G_{i,j} \) in \( \theta_r \): setting \( (y_1,y_2,\ldots,y_m) = t \) fixes the arguments of \( F_j \) or \( G_{i,j} \) in this particular occurrence at \( t' \), say (notice that if the occurrence is \( F_j \) then \( t' \) has length \( m+1 \), otherwise it has length \( m \)): we define that:

\[
\begin{align*}
&u_{ij}(t') \in D_r(t) \iff F_j(t') \text{ appears in } \theta_r(t); \\
&\neg u_{ij}(t') \in D_r(t) \iff \neg F_j(t') \text{ appears in } \theta_r(t); \\
v_{ij}(t') \in D_r(t) \iff G_{i,j}(t') \text{ appears in } \theta_r(t); \\
&\neg v_{ij}(t') \in D_r(t) \iff \neg G_{i,j}(t') \text{ appears in } \theta_r(t).
\end{align*}
\]

(All variables introduced above are distinct.)

Suppose that:

\[
S \models \forall x_1 \exists z_1 F_1(x_1,z_1) \land \ldots \land \forall x_m \exists z_m F_m(x_m,z_m) \land \forall y_1 \forall y_2 \ldots \forall y_m \psi
\]

via the set of relations \( \{F_j,G_{i,j} : i = 1,2,\ldots,k; j = 1,2,\ldots,m\} \) (notice that we do not distinguish between relations and relation symbols, as the denotation is always clear from the context). Consider the following truth assignment \( f \) on the Boolean variables of \( \sigma(S) \): for all \( i = 1, 2, \ldots, k, j = 1, 2, \ldots, m, s \in |S|, \) and \( t \in |S|^m \):

\[
\begin{align*}
f(w_j) &= True; \\
f(u_{ij}(t,s)) &= True \iff F_j(t,s) \text{ holds}; \\
f(v_{ij}(t)) &= True \iff G_{i,j}(t) \text{ holds}.
\end{align*}
\]

Clearly, \( f \) satisfies any clause \( B_r \) and \( C_r(t), \) and any clause \( D_r(t) \) where:

\[
(S,t) \models \chi_r(y_1,y_2,\ldots,y_m).
\]

Suppose that \( t \) and \( r \) are such that:

\[
(S,t) \models \neg \chi_r(y_1,y_2,\ldots,y_m).
\]
Then $(S,t) \vdash \theta_r(y_1,y_2,...,y_m)$ (and in particular $\theta_r$ must be non-empty). So, for some $i, j$, $F_j(t')$ (resp. $\neg F_j(t')$) appears in $\theta_r(t)$ and $F_j(t')$ (resp. $\neg F_j(t')$) holds, or $G_{i,j}(t')$ (resp. $\neg G_{i,j}(t')$) appears in $\theta_r(t)$ and $G_{i,j}(t')$ (resp. $\neg G_{i,j}(t')$) holds. In any event, some literal of the clause $D_r(t)$ is given the value $True$ by $f$ and so $f$ is a satisfying truth assignment for $\sigma(S)$.

Conversely, suppose that $f$ is a satisfying truth assignment for $\sigma(S)$. Define the set of relations $\{F_{i,j},G_{i,j} : i = 1,2,...,k; j = 1,2,...,m\}$ as follows: for all $i = 1, 2, ..., k$, $j = 1, 2, ..., m$, $s \in |S|$, and $t \in |S|^m$:

$F_{j}(t,s)$ holds $\iff f(u_{j}(t,s)) = True$;

$G_{i,j}(t)$ holds $\iff f(v_{i,j}(t)) = True$

(notice that $f(w_j) = True$, for each j). As any clause $D_r(t)$ of $\sigma(S)$ is satisfied by $f$, then clearly $(S,t) \vdash \psi_r(y_1,y_2,...,y_m)$. Also, as any clause $C_r(t)$ is satisfied by $f$, then $f(u_{i}(t,s)) = True$, for some $s \in |S|$; that is, $F_{r}(t,s)$ holds. Consequently:

$S \models \forall x_1 \exists z_1 F_1(x_1,z_1) \land ... \land \forall x_m \exists z_m F_m(x_m,z_m) \land \forall y_1 \forall y_2 ... \forall y_m \psi$

via the set of relations $\{F_{i,j},G_{i,j} : i = 1,2,...,k; j = 1,2,...,m\}$.

For any set $R$ of relations $\{F_{i,j},G_{i,j} : i = 1,2,...,k; j = 1,2,...,m\}$, let $\sigma(R)$ denote the corresponding truth assignment (as above) on the variables of $\sigma(S)$, and for any truth assignment $f$ on the variables of $\sigma(S)$, let $\sigma^{-1}(f)$ denote the corresponding set of relations $\{F_{i,j},G_{i,j} : i = 1,2,...,k; j = 1,2,...,m\}$ (this notation makes sense as we insist that $\sigma(R)(w_j) = True$, for each j). Hence, for any structure $S \in STRUCT(\tau(\Omega))$:

$S \models \forall x_1 \exists z_1 F_1(x_1,z_1) \land ... \land \forall x_m \exists z_m F_m(x_m,z_m) \land \forall y_1 \forall y_2 ... \forall y_m \psi$

via the set of relations $R$ if and only if:

$\sigma(R)$ is a satisfying truth assignment for $\sigma(S)$.

For any $S \in STRUCT(\tau(\Omega))$, partition the set of variables of $\sigma(S)$ into the sets:

$Y_i = \{v_{i,j}(t) : j = 1,2,...,m; t \in |S|^m\}$, for each $i = 1, 2, ... , k$, and:

$Z = \{u_{i}(t,s) : j = 1,2,...,m; s \in |S|; t \in |S|^m \cup \{w_j : j = 1,2,...,m\}\}.$

Then clearly $S \models \phi$ if and only if $\exists Y_1 \forall Y_2 ... \exists Y_k \exists Z[\sigma(S) = True]$. Hence, if we can show that the translation $\sigma$, together with the partition $Y_1, Y_2, ..., Y_k \cup Z$, can be described by projective formulae, then we clearly have that CNFk is complete for $\Sigma_k^p$ via projection translations whenever k is odd.

By [Fag74] and [Sto77], $\Pi_k^p$ coincides with the class of problems expressible by a sentence $\phi$ of second-order logic of the form:

$\forall G_{1,1}... \forall G_{1,n_1} \exists G_{2,1}... \exists G_{2,n_2} ... Q_k G_{k,1}... Q_k G_{k,n_k} \phi$,

where each $G_{i,j}$ is a relation symbol of arity $a_{i,j}$, $Q_k$ is the quantifier $\forall$ (resp. $\exists$) if $k$ is odd (resp. even), and $\phi$ is a first-order formula. By proceeding as above, we may clearly assume that $\phi$ is of the form:

$\forall G_{1,1}... \forall G_{1,n_1} \exists G_{2,1}... \exists G_{2,n_2} ... Q_k G_{k,1}... Q_k G_{k,n_k} \exists F_1 ... \exists F_m$
\[ \forall x_1 \exists z_1 F_1(x_1, z_1) \land \ldots \land \forall x_m \exists z_m F_m(x_m, z_m) \land \forall y_1 \forall y_2 \ldots \forall y_r \psi, \]

with the notation as before. For \( k \) even, again by proceeding as above, we can obtain a translation \( \sigma \) such that for any \( S \in \text{STRUCT}(\tau(\emptyset)) \), \( \sigma(S) \) is a Boolean formula in c.n.f. and there is a partition of the variables involved in \( \sigma(S) \) into disjoint sets \( Y_1, Y_2, \ldots, Y_k \), with the property that:

\[ S \models \emptyset \iff \forall Y_1 \exists Y_2 \ldots \exists Y_k \exists z[\sigma(S) = \text{True}]. \]

Hence, \( S \models \emptyset \iff \neg \exists Y_1 \forall Y_2 \ldots \forall Y_k \forall z[\sigma(S) = \text{False}] \). It is easy to see that there is a Boolean formula \( \sigma'(S) \) in d.n.f. such that \( \sigma(S) = \text{False} \) if and only if \( \sigma'(S) = \text{True} \), and that this formula \( \sigma'(S) \) can be described in terms of \( \sigma(S) \) via monotone projective formulae. Thus, assuming that the translation \( \sigma \), together with the partition \( Y_1, Y_2, \ldots, Y_k \cup \emptyset \), can be described by projective formulae, co-\( \text{DNF}_k \) is clearly complete for \( \Pi_k^p \) via projection translations; that is, \( \text{DNF}_k \) is complete for \( \Sigma_k^p \) via projection translations (whenever \( k \) is even).

It remains to show that the translation \( \sigma \) and the partition, above, can be described by projective formulae: we do not give the full descriptions but merely sketch how it can be done. For any relevant structure \( S \) of size \( n \), our structure \( \sigma(S) \) is to be of size \( n^{2m+3+k} \):

for a Boolean variable:

components 1 and 2 of some tuple of variables over \( \{0, 1, \ldots, n-1\} \) refer to the Boolean variable type, namely \( u, v, \) or \( w \), with \( u \)- (resp. \( v \)-, \( w \)-) type variables having these components set at \( (0, 0) \) (resp. \( (0, \text{max}) \), \( (\text{max}, 0) \));

components 3, 4, ..., \( k+m+2 \) refer to the subscript of the Boolean variable, with subscript \( j \) corresponding to the \( (k+m) \)-tuple with all components 0 except for the \( (k+m+1) \)-th which is \( \text{max} \), and subscript \( (i,j) \) corresponding to the \( (k+m) \)-tuple with all components 0 except for the \( i \)-th and the \( (k+m+1) \)-th which are \( \text{max} \);

components \( k+m+3, k+m+4, \ldots, k+2m+3 \) refer to the argument of the Boolean variable in the obvious way (Boolean variables with arguments of length \( m \) have the last component set at 0);

for a clause:

components 1 and 2 of some tuple of variables over \( \{0, 1, \ldots, n-1\} \) refer to the clause type, namely \( B, C \) or \( D \), with \( B \)- (resp. \( C \)-, \( D \)-) type clauses having these components set at \( (0, 0) \) (resp. \( (0, \text{max}) \), \( (\text{max}, 0) \));

components 3, 4, ..., \( k+m+2 \) refer to the subscript of the clause (as above);

components \( k+m+3, k+m+4, \ldots, k+2m+3 \) refer to the argument of the clause (as above)
(the description of the partition should be obvious given how we describe variables). For example, the projective formula describing the fact that $C_1(t) = \{u_1(t,0), u_1(t,1), \ldots, u_1(t,n-1)\}$ is as follows:

\[
(x_1, x_2) = 0 \land (y_1, y_2) = (0, \max) \land (x_3, \ldots, x_{k+2}) = (y_3, \ldots, y_{k+2}) = 0 \land \\
(x_k + 3 = y_k + 3 = \max) \land (x_{k+4}, \ldots, x_{k+m+2}) = (y_{k+4}, \ldots, y_{k+m+2}) = 0 \land \\
(x_{k+m+3}, \ldots, x_{k+2m+2}) = (y_{k+m+3}, \ldots, y_{k+2m+2}) \land y_{k+2m+3} = 0,
\]

where $x = (x_1, x_2, \ldots, x_{k+2m+3})$ and $y = (y_1, y_2, \ldots, y_{k+2m+3})$: the other projective formulae can be defined similarly. (The reader is referred to many of the proofs of [Ste91c], for example, for similar descriptions.) □

As mentioned earlier, $3\text{CNF}_k$ (resp. $3\text{DNF}_k$) is complete for $\Sigma_k^p$ via logspace reductions when $k$ is odd (resp. even). The proof of Theorem 3.1 does not appear to be modifiable so that similar results hold via projection translations, essentially because of the construction of the clause $D_r(t)$ and stipulations like:

\[
\{w_q, \neg w_q\} \subseteq D_r(t) \iff (S, t) \models x_{r_q}(y_1, y_2, \ldots, y_m)
\]

(the notation is as in the proof). In order to be able to describe such stipulations using projective formulae, we needed to split $x_r$ into its constituents (viz. the $x_{r_q}$'s) and use different Boolean variables for each constituent (viz. the $w_q$'s); but doing so may introduce many literals into the clause $D_r(t)$. However, all is not lost.

**Theorem 3.2.** There exists a monotone projection translation from $\text{CNF}_k$ (resp. $\text{DNF}_k$) to $3\text{CNF}_k$ (resp. $3\text{DNF}_k$), and so $3\text{CNF}_k$ (resp. $3\text{DNF}_k$) is complete for $\Sigma_k^p$ via projection translations when $k$ is odd (resp. $k$ is even).

**Proof.** Let $S$ be an instance of size $n$ of the problem $\text{CNF}_k$, for some odd $k$. The literals involved in the instance $S$ are $X_0, X_1, \ldots, X_{n-1}$, $\neg X_0, \neg X_1, \ldots, \neg X_{n-1}$; that is, $L_0, L_1, \ldots, L_{2n-1}$, respectively. An instance $\sigma(S)$ of $3\text{CNF}_k$ is defined as follows:

(i) for each $j = 0, 1, \ldots, n-1$, replace the $j$th clause $C_j$ of $S$ with the clauses $C_{j,1}, C_{j,2}, \ldots, C_{j,2n-2}$, where for each $i = 1, 2, \ldots, 2n-2$:

- $L_i \in C_{j,i} \iff L_i \in C_j$;
- $L_0 \in C_{j,1} \iff L_0 \in C_j$;
- $L_{2n-1} \in C_{j,2n-2} \iff L_{2n-1} \in C_j$;
- $Z_{j,i} \in C_{j,i}$ if $i \neq 2n-2$;
- $Z_{j,i,1} \in C_{j,i}$ if $i \neq 1$;

($Z_{j,1}, Z_{j,2}, \ldots, Z_{j,2n-3}$ are new, distinct Boolean variables);

(ii) for each $j = 1, 2, \ldots, k$, replace the subset of variables $Y_j$, given by $M_j^S$, by the subset $Y'_j$, where:

$$Y'_j = Y_j \text{ if } j \neq k;$$

-13-
\[ Y_k' = Y_k \cup \{Z_{j,i} : j = 0,1,\ldots,n-1; i = 1,2,\ldots,2n-3\}. \]

(This is the usual reduction of a c.n.f. formula to one with at most three literals per clause [GJ79]). It is easy to see that for any truth assignment \( t \) on the Boolean variables involved in the c.n.f. formula of \( S \):

- \( t \) is satisfying \( \iff \) \( t \) can be extended to a satisfying truth assignment \( t' \) on the Boolean variables involved in the c.n.f. formula of \( \sigma(S) \).

Hence, it is easy to see that \( S \in \text{CNF}_k \) if and only if \( \sigma(S) \in \text{3CNF}_k \). As the reduction from \( S \) to \( \sigma(S) \) can clearly be described by a monotone projective formula, then, by Lemma 3.1 of [Ste91c], \( \text{3CNF}_k \) is complete for \( \Sigma_k^p \) via projection translations, when \( k \) is odd.

When \( k \) is even, that \( \text{3DNF}_k \) is a monotone projection of \( \text{DNF}_k \) follows similarly and so, by Lemma 3.1 of [Ste91c], \( \text{3DNF}_k \) is complete for \( \Sigma_k^p \) via projection translations. \( \square \)

We add that it is unknown whether the versions of \( \text{3CNF}_k \) and \( \text{3DNF}_k \) where each clause has exactly 3 literals are complete for \( \Sigma_k^p \) via projection translations, when \( k \) is odd and \( k \) is even, respectively. Whilst similar statements are true in the case of logspace reductions, it should be pointed out that in [Ste91c] it was shown that \( \preceq \text{3SAT} \) is complete for \( \text{NP} \) via projection translations but only that \( \text{3SAT} \) is complete for \( \text{NP} \) via iterated projection translations: whether \( \text{3SAT} \) is complete for \( \text{NP} \) via projection translations is unknown.

4. Boolean labelled structures.

In this section, we show that certain problems are complete for certain classes at the bottom of the Polynomial Hierarchy via projection translations: the corresponding decision problems were not even known to be complete for the respective classes via polynomial-time reductions. Our techniques depend on various logics possessing normal forms.

We begin by considering networks of processors where the links between processors might fail and where these failures may possibly be dependent on one another. Reliability in such networks (that is, the consideration of whether a network works under different failure scenarios) has not (to our knowledge) been much considered from a complexity-theoretic point of view: only where the link failures are independent of one another have results been obtained (the related case where processors as opposed to links fail has also been studied). For example:

- (i) the problem of determining the probability that there is a path in a (directed or undirected) network between two distinguished processors, given that each link
fails with independent probability $p$, is $\#P$-complete ([Val79]): in fact, this is true for undirected and acyclic directed planar graphs having vertex degree at most 3 ([Pro86]);

(ii) the problem of determining the probability that there is a path in a (directed or undirected) network between a distinguished processor and every other processor, given that each link fails with independent probability $p$, is $\#P$-complete ([Hag80], [Jer81], [PB83]): in fact, even approximating this probability to within a given $\varepsilon$ is $\#P$-complete ([PB83]);

(iii) the problem of determining the probability that the surviving processors in a (directed or undirected) network whose processors fail and whose links are safe are non-empty and connected is $\#P$-complete: this is still true when the network is a split, planar, or bipartite graph ([SSS91]).

The following result involves networks with dependent link failures:

(iv) the decision problem an instance of which consists of:

- a network of processors $G = (V, E)$, a subset of processors $V' \subseteq V$, a bound $B$,
- and for each (undirected) link $e \in E$ a lower bound $a(e)$ and an upper bound $b(e)$ on its failure probability,

and a yes-instance of which consists of an instance where:

- there is an assignment of a probability to each subset $E' \subseteq E$ (representing the probability that the edges in $E'$ fail and those in $E \setminus E'$ do not) such that for each edge $e$ the cumulative failure probability for $e$ lies between $a(e)$ and $b(e)$, and such that the probability that all processors in $V'$ remain connected to each other is greater than $B$,

is $\text{NP}$-hard ([Zem82]).

Other similar completeness results can be found in [GJ79] and [Joh85].

Our networks are directed. We also require that the interdependencies can be described by Boolean literals. By this we mean that each link in a network of size $n$ is labelled by a Boolean literal over the set of Boolean variables $\{X_{i,j} : i,j = 0,1,\ldots,n-1\}$ or $\text{True}$, and for any given truth assignment on these Boolean variables, a link succeeds if and only if its label is $\text{True}$ under the truth assignment. Consequently, the different truth assignments correspond to the different possible failure scenarios.

Such a set-up is not as unrealistic as it may seem, for consider the perfectly acceptable interdependency relationship:

"link $(a,b)$ fails if and only if link $(c,d)$ fails, and link $(a,b)$ fails if and only if link $(e,f)$ succeeds".

In order to translate this relationship into the above form, we have the link $(a,b)$ labelled with a Boolean variable $X$, the link $(c,d)$ also labelled with $X$, and the link
(e,f) labelled with \( \neg X \). Consequently, whether X is true or not determines the difference possibilities for the failure of these links. Whilst our notion of failure dependency appears to be very restricted, we shall see that certain natural problems concerning the reliability of such networks are seemingly hard. Our problems do not involve computing the probability that a network satisfies some property but merely deciding whether the probability that a network satisfies some property is 1: when the link failures are independent of one another these problems become relatively trivial. Henceforth, we choose to consider digraphs, vertices, and edges as opposed to networks, processors, and links.

Consider the decision problem DETERMINISTIC RELIABILITY OF NETWORKS WITH BOOLEAN-RELATED LINK FAILURES defined as follows:

Instance of size n : a digraph G on the vertices \{0,1,...,n-1\} such that every edge is labelled with a literal over the set of Boolean variables \( \{X_{i,j} : i,j = 0,1,...,n-1\} \) or \( True \);

Yes-instance of size n : an instance of size n such that for every truth assignment \( t \) on \( \{X_{i,j} : i,j = 0,1,...,n-1\} \), there is always a deterministic path from the vertex 0 to the vertex \( n-1 \) in the digraph \( t(G) \), which is the digraph on the vertices \( \{0,1,...,n-1\} \) with those edges of G set at \( True \) under \( t \).

We choose to encode the above decision problem over the vocabulary \( \tau^* \) consisting of the relation symbols E and R, of arities 2 and 5, respectively: we call this encoding DRELNETBLF. A structure \( S \in \text{STRUCT}(\tau^*) \) of size n describes an instance of the above problem as follows:

(i) there is an edge \((x,y)\) in \( G^S \iff (S,x,y) \models E(x,y); \)
(ii) the edge \((x,y)\) of \( G^S \) is labelled \( True \iff \)
\[(S,x,y) \models R(x,y,0,0,0) \land R(x,y,\max,\max,\max) \land \forall u \forall v \forall w[R(x,y,u,v,w) \Rightarrow (u,v,w) = (0,0,0) \lor (u,v,w) = (\max,\max,\max)];\]
(iii) the edge \((x,y)\) of \( G^S \) is labelled with the literal \( X_{u,v} \iff \)
\[(S,x,y) \models R(x,y,u,v,0) \land \forall u_1 \forall v_1 \forall w[R(x,y,u_1,v_1,w) \Rightarrow (u_1,v_1,w) = (u,v,0)];\]
(iv) the edge \((x,y)\) of \( G^S \) is labelled with the literal \( \neg X_{u,v} \iff \)
\[(S,x,y) \models R(x,y,u,v,\max) \land \forall u_1 \forall v_1 \forall w[R(x,y,u_1,v_1,w) \Rightarrow (u_1,v_1,w) = (u,v,\max)].\]

We discuss this encoding scheme after proving the following result.

**Theorem 4.1.** DRELNETBLF is complete for co-NP via projection translations.

**Proof.** By [Fag74], any problem \( \Omega \in \text{co-NP} \) can be represented by a second-order sentence of the form:

- 16 -
∀R₁ ∀R₂ ... ∀Rₘφ,
where φ is a first-order sentence (with successor) and each Rᵢ is a relation symbol
of arity aᵢ. By [Imm87], the sentence φ is equivalent to a sentence of DTC¹[FO ≤]
of the form:

DTC[λxyψ(x,y)](0,max),

where ψ(x,y) is a projective formula and x and y are k-tuples of variables. Hence,
Ω can be represented by a second-order sentence of the form:

∀R₁ ∀R₂ ... ∀RₘDTC[λxyψ(x,y)](0,max);

we may clearly assume that all relation symbols have arity a and that m+a = k
(the reasons for doing so will soon become apparent).

Let the vocabulary τ' be the vocabulary τ(Ω) augmented with the relation
symbols R₁, R₂, ..., Rₘ. Then, for any structure S ∈ STRUCT(τ(Ω)):

S ∈ Ω ⇔ for all structures S' ∈ STRUCT(τ') such that S'|τ(Ω) = S, we have that:

S' ⊨ DTC[λxyψ(x,y)](0,max).

Now, we may assume that ψ(x,y) is of the form:

(α₁ ∧ β₁) ∨ (α₂ ∧ β₂) ∨ ... ∨ (αₚ ∧ βₚ),

for some p, where:

(i) each αᵢ is a conjunction of logical atomic relations, s, =, and their negations;
(ii) each βᵢ is atomic or negated atomic and:

βᵢ(x,y) does not involve any of the symbols R₁, R₂, ..., Rₘ, for i = 1, 2, ..., q;

βᵢ₊₁(x,y) = Rᵢ(zᵢ), for some tuple of variables zᵢ, for i = 1, 2, ..., r;

βᵢ₊₁₊₁(x,y) = ¬Rᵢ₊₁(zᵢ₊₁), for some tuple of variables zᵢ₊₁, for i = 1, 2, ..., r,

with q + r + t = p;

(iii) if i ≠ j, then αᵢ and αⱼ are mutually exclusive.

Consider the following projective formulae involving the variables of the
k-tuples x, y, u, v, and w, where u = (u₁,u₂) with |u₁| = m and |u₂| = a;

ψE(x,y) = [α₁(x,y) ∧ β₁(x,y)] ∨ ... ∨ [αᵢ(x,y) ∧ βᵢ(x,y)] ∨ αᵢ₊₁(x,y) ∨ ...

... ∨ αₚ(x,y);

ψF(x,y,u,v,w) = [α₁(x,y) ∧ β₁(x,y) ∧ u = v = w = 0] ∨ ...

... ∨ [αᵢ(x,y) ∧ βᵢ(x,y) ∧ u = v = w = 0]

∨ [α₁(x,y) ∧ β₁(x,y) ∧ u = v = w = max] ∨ ...

... ∨ [αᵢ(x,y) ∧ βᵢ(x,y) ∧ u = v = w = max];

ψL(x,y,u,v,w) = [αᵢ₊₁(x,y) ∧ (u₁,u₂) = (#c₁,z₁) ∧ v = 0 ∧ w = 0] ∨ ...

... ∨ [αᵢ₊₁(x,y) ∧ (u₁,u₂) = (#cᵢ,zᵢ) ∧ v = 0 ∧ w = 0]

∨ [αᵢ₊₁₊₁(x,y) ∧ (u₁,u₂) = (#cᵢ₊₁,zᵢ₊₁) ∧ v = 0 ∧ w = max] ∨ ...

... ∨ [αᵢ₊₁₊₁₊₁(x,y) ∧ (u₁,u₂) = (#cᵢ₊₁+zᵢ₊₁) ∧ v = 0 ∧ w = max];

#c denotes the tuple with the cᵢth entry set at max and every other entry set at 0.

Given a structure S ∈ STRUCT(τ(Ω)), the formula ψE describes which edges
appear in a digraph $G^S$ on the vertices \( \{ x : x \in |S|^k \} \); the formula \( \psi_T \) describes those edges of $G^S$ which are labelled True; and the formula \( \psi_L \) describes the labels of those edges of $G^S$ that are not labelled True: notice that every edge of the digraph $G^S$ has exactly one label and only edges of $G^S$ have labels.

Denote the projection translation of $S$ with respect to $\psi_E$ and $\psi_T \lor \psi_L$ (which is indeed equivalent to a projective formula) by $\sigma(S)$. We claim that:

$$S \in \Omega \iff \sigma(S) \in \text{DRELNETBLF}.$$ 

We begin by noting that, for any structure $S \in \text{STRUCT}(\tau(\Omega))$, the Boolean variable involved in the labelling of the edge $(x,y)$ of the digraph $G^S$ of $\sigma(S)$ is $X_{u,v}$, where \( u = (u_1,u_2) = (#c_i,z_i) \), for some $i = 1, 2, \ldots, q+r$, and $v = 0$: associate the truth or falsity of this Boolean variable with the truth or falsity of $R_{c_i}(z_i)$, and vice versa.

Suppose that $S \in \Omega$. Then for any set of relations $R_1, R_2, \ldots, R_m$:

$$(S,R_1,R_2,\ldots,R_m) \models \text{DTC}(\lambda xy \psi(x,y))[0,\text{max}):$$

let the digraph obtained from $\psi(x,y)$ with these relations be denoted by $G^S(R_1,R_2,\ldots,R_m)$. By the above association, we obtain a truth assignment $t$ on the Boolean variables involved in the labelling of $G^S$, and it should be clear that any extension of this truth assignment, also denoted by $t$, to all the available Boolean variables is such that:

the digraph $G^S(R_1,R_2,\ldots,R_m)$ is identical to the digraph $t(G^S)$. Consequently, if $S \in \Omega$ then $\sigma(S) \in \text{DRELNETBLF}$. The converse is similar using the reverse association described above, and so the result follows. □

Notice that the encoding scheme used to encode DETERMINISTIC RELIABILITY OF NETWORKS WITH BOOLEAN-RELATED LINK FAILURES as the problem DRELNETBLF is such that not all structures over $\tau^*$ are encodings of instances of the underlying decision problem and there may be different encodings, as structures over $\tau^*$, of the same instance of the underlying decision problem. However, our encoding scheme is "reasonable" in the sense of [GJ79]. For instance, given a structure over $\tau^*$, there is a first-order sentence which detects whether it is the encoding of an instance of the underlying decision problem, and, if so, there are quantifier-free first-order formulae which answer "fundamental" questions about the underlying instance, e.g. "is the label of the edge $(x,y)$ True?". Also, there is a first-order sentence that determines when two structures over $\tau^*$ are encodings of the same underlying instance.

We mention these points here as, according to the results of [Ste91c], it appears that how we encode problems seems to matter when we are considering such weak reductions as projection translations. For example, from [Ste91c], we can show
that \( \leq 3\text{SAT} \) is complete for \( \text{NP} \) via projection translations but it is unknown whether \( 3\text{SAT} \) is. As to what constitutes an "acceptable" or "reasonable" encoding scheme when viewing problems as sets of finite structures over some vocabulary and considering very weak reductions, we have yet to formulate (and do not do so in this paper).

Consider the related decision problem RELIABILITY OF NETWORKS WITH BOOLEAN-RELATED LINK FAILURES, defined as follows:

Instance of size \( n \) : a digraph \( G \) on the vertices \( \{0,1,\ldots,n-1\} \) such that every edge is labelled with a literal over the set of Boolean variables \( \{X_{i,j} : i,j = 0,1,\ldots,n-1\} \) or \( \text{True} \);

Yes-instance of size \( n \) : an instance of size \( n \) such that for every truth assignment \( t \) on \( \{X_{i,j} : i,j = 0,1,\ldots,n-1\} \), there is always a path from the vertex 0 to the vertex \( n-1 \) in the digraph \( t(G) \), which is the digraph on the vertices \( \{0,1,\ldots,n-1\} \) with those edges of \( G \) set at \( \text{True} \) under \( t \).

We encode the above decision problem over the vocabulary \( \tau^* \) to obtain RELNETBLF as we did previously.

**Theorem 4.2.** RELNETBLF is complete for co-NP via projection translations.

**Proof.** Almost identical to the proof of Theorem 4.1 except that we use the logic \( \text{TC}^1[\text{FO}_\leq] \) in place of \( \text{DTC}^1[\text{FO}_\leq] \), together with the appropriate results from [Imm87]. \( \square \)

When we consider Hamiltonian paths in networks, we find that the situation is slightly different. Define the decision problem HAMILTONIAN RELIABILITY OF NETWORKS WITH BOOLEAN-RELATED LINK FAILURES as follows:

Instance of size \( n \) : a digraph \( G \) on the vertices \( \{0,1,\ldots,n-1\} \) such that every edge is labelled with a literal over the set of Boolean variables \( \{X_{i,j} : i,j = 0,1,\ldots,n-1\} \) or \( \text{True} \);

Yes-instance of size \( n \) : an instance of size \( n \) such that for every truth assignment \( t \) on \( \{X_{i,j} : i,j = 0,1,\ldots,n-1\} \), there is always a Hamiltonian path from the vertex 0 to the vertex \( n-1 \) in the digraph \( t(G) \), which is the digraph on the vertices \( \{0,1,\ldots,n-1\} \) with those edges of \( G \) set at \( \text{True} \) under \( t \).

We encode the above decision problem over the vocabulary \( \tau^* \) to obtain HRELNETBLF as we have done previously.
Theorem 4.3. \textit{HRELNETBLF} is complete for $\Pi^p_2$ via projection translations.

Proof. By [Fag74], any problem in $\Pi^p_2$ can be represented by a sentence of second-order logic of the form:
$$\forall R_1 \forall R_2 \ldots \forall R_m \exists T_1 \exists T_2 \ldots \exists T_p \phi,$$
and any problem in $\text{NP}$ can be represented by a sentence of second-order logic of the form:
$$\exists T_1 \exists T_2 \ldots \exists T_p \phi,$$
where each $R_i$ and $T_j$ is a relation symbol and $\phi$ is a first-order sentence. Consequently, by [Ste91a], any problem in $\Pi^p_2$ can be represented by a sentence $\phi$ of the form:
$$\forall R_1 \forall R_2 \ldots \forall R_m \text{HP}[\lambda x y \psi(x,y)](0, \text{max}),$$
where $x$ and $y$ are $k$-tuples of variables and $\psi(x,y)$ is a projective formula. The proof now proceeds exactly as did that of Theorem 4.1, and the result follows. \qed

We mention here that the above problems concerning reliability in networks can be looked at in a different light. In [GMO76] the following decision problem was considered, called FORBIDDEN PAIRS:

Instance of size $n$ : a digraph $G$ on the vertices $\{0, 1, \ldots, n-1\}$ and a collection $C$ of (forbidden) pairs of edges;

Yes-instance of size $n$ : an instance $G$ of size $n$ such that there is a path from the vertex 0 to the vertex $n-1$ containing at most one edge from each pair of $C$.

It was shown that FORBIDDEN PAIRS is $\text{NP}$-complete via logspace reductions (as are the variants where $C$ consists of forbidden pairs of vertices, or where $G$ is acyclic with no vertex of in- or out-degree exceeding 2). It is easy to see that any instance $G$, of size $n$, of RELIABILITY OF NETWORKS WITH BOOLEAN-RELATED LINK FAILURES can be considered as an instance of FORBIDDEN PAIRS, and that if we merely insist that there should exist a truth assignment $t$ such that $t(G)$ has a path from vertex 0 to vertex $n-1$, instead of insisting that this should be so for all truth assignments $t$ as we did with RELIABILITY OF NETWORKS WITH BOOLEAN-RELATED LINK FAILURES, then we (essentially) obtain the decision problem FORBIDDEN PAIRS. It is not hard to see (using the fact that $(\pm \text{TC})^*[\text{FO}_\leq] = \text{TC}^1[\text{FO}_\leq]$ [Imm88]) that RELIABILITY OF NETWORKS WITH BOOLEAN-RELATED LINK FAILURES is (essentially) the complementary decision problem of FORBIDDEN PAIRS.
Consequently, it turns out that the decision problem RELIABILITY OF NETWORKS WITH BOOLEAN-RELATED LINK FAILURES has already been considered from a complexity-theoretic point of view, for it was shown to be complete for NP via logspace reductions in [GMO76]: the problems DETERMINISTIC RELIABILITY OF NETWORKS WITH BOOLEAN-RELATED LINK FAILURES and HAMILTONIAN RELIABILITY OF NETWORKS WITH BOOLEAN-RELATED LINK FAILURES have not been considered before in any guise. We also remark that the versions of the network reliability problems studied above where the digraphs are acyclic and do not have vertices with in- or out-degree exceeding 2 or where the vertices are labelled with Boolean literals, are complete for NP via projection translations (as the formula $\psi$ in the proofs of Theorems 4.1 (4.2) and 4.3 can be modified accordingly).

We choose to consider Boolean labelled networks above, as opposed to digraphs with forbidden pairs, for two reasons; firstly, problems involving network reliability can not be naturally formulated in the "forbidden pair" context (that is, problems where we wish to say "for all truth assignments ..." as opposed to those where we wish to say "there exists a truth assignment ..."), and secondly, other structures, apart from digraphs, can easily be labelled with Boolean literals. For example, we might have a c.n.f. Boolean formula where the literals of the clauses are themselves labelled with Boolean literals (over a disjoint set of Boolean variables) or True. Consider the following decision problem LABELLED CNF BOOLEAN FORMULA:

**Instance of size n:** a Boolean formula $BF$ in c.n.f. over the set of Boolean variables $\{X_i : i = 0,1,\ldots,n-1\}$ where each literal of each of at most $n$ clauses is labelled with a Boolean literal over the set of Boolean variables $\{W_{ij} : i,j = 0,1,\ldots,n-1\}$ or True;

**Yes-instance of size n:** an instance of size $n$ such that for every truth assignment $t$ on $\{W_{ij} : i,j = 0,1,\ldots,n-1\}$, the resulting Boolean c.n.f. formula $t(BF)$ obtained from $BF$ by only including those literals over $\{X_i : i = 0,1,\ldots,n-1\}$ whose labels are set at True under $t$ is satisfiable.

We encode this decision problem as expected over the vocabulary consisting of the 4 relation symbols $P$, $N$, $L_p$, and $L_n$ of arities 2, 2, 5, and 5, respectively ($L_p$ (resp. $L_n$) describes the labels of the positive (resp. negative) literals of the clauses), and call this encoding LABCNFBF.

Let $BF$ be a labelled c.n.f. Boolean formula (as above), let $Y_1 = \{W_{ij} : i,j = 0,1,\ldots,n-1\}$, and let $Y_2 = \{X_i : i = 0,1,\ldots,n-1\}$. Define the Boolean formula $BF'$ as follows:
(i) for each non-empty clause $C$ of $BF$, there is a Boolean formula $C'$, and $BF'$ is the conjunction of all such formulae $C'$;

(ii) $C'$ is the disjunction of conjunctions defined as follows:
if the literal $L_2$ over $Y_2$, labelled $L_1$ (over $Y_1$), is in clause $C$ of $BF$ then:
\[ \neg L_1 \land L_2 \] is a conjunction in $C'$.

The Boolean formula $BF'$ can easily be expanded so that it is in c.n.f., and so we can consider $BF'(Y_1,Y_2)$ as an instance of co-DNF$_2$. It is not hard to see that $BF$, with its labelling, is a yes-instance of LABCNFBF if and only if $BF'(Y_1,Y_2)$ is a yes-instance of co-DNF$_2$, and that the above reduction can be described by monotone projective formulae.

Just as we showed the problems DRENBLF, RELNETBLF, and HRELNETBLF to be complete for co-NP, co-NP, and II$^2_P$, respectively, so we can easily show LABCNFBF to be complete for II$^2_P$ (by proceeding exactly as we have done throughout this section). Hence, by Lemma 3.1 of [Ste91c], co-DNF$_2$ is complete for II$^2_P$ via projection translations, and so DNF$_2$ is complete for II$^2_P$ via projection translations (without successor, in fact). Consequently, we have another proof that DNF$_2$ is complete for II$^2_P$ via projection translations. By considering appropriate problems involving labelled Boolean formulae, it is not hard to see that the results of Theorem 3.1 could also have been established in this way. We remark that the logspace completeness of the decision problems involved in the statement of Theorem 3.1 was established by entirely different methods.

Just as we can have Boolean labelled digraphs and c.n.f. formulae, so we can clearly have Boolean labelled graphs. However, the situation here is not as straightforward as it is for digraphs because of how graphs are encoded as sets of structures over vocabularies. Let $S$ be a structure over $\tau_2$. Then there is an edge in the graph corresponding to $S$ between the vertex $x$ and the vertex $y$, say, if and only if $E^S(x,y)$ or $E^S(y,x)$ holds.

Let the decision problem CERTAIN 3-COLOURING OF BOOLEAN EDGE-LABELLED GRAPHS be defined as follows:

Instance of size $n$: a graph $G$ on the vertices $\{0,1,...,n-1\}$ such that every edge is labelled with a disjunction of two literals where each literal is over the set of Boolean variables $\{X_{i,j} : i,j = 0,1,...,n-1\}$ or is $True$;

Yes-instance of size $n$: an instance $G$ of size $n$ such that for every truth assignment $t$ on $\{X_{i,j} : i,j = 0,1,...,n-1\}$, the graph $t(G)$, obtained from $G$ by including only those edge set at $True$ under $t$, can be coloured with 3 colours.
This decision problem is encoded over the vocabulary $\tau^*$, consisting of the relation symbols $E$ and $R$ of arities 2 and 5, respectively, as usual, and this encoding is called CERT3COL.

**Theorem 4.4.** CERT3COL is complete for $\Pi_2^p$ via projection translations.

**Proof.** By [Ste91b] and [Ste91c], the problem 3COL is complete for $\text{NP}$ via projection translations, and so just as in the proof of Theorem 4.3, any problem in $\Pi_2^p$ can be represented by a sentence of the form:

$$\forall R_1 \forall R_2 \ldots \forall R_m 3\text{COL}[\lambda xy \psi(x,y)](0, \text{max}),$$

where the $R_i$ are relation symbols, $x$ and $y$ are $k$-tuples of variables, and $\psi(x,y)$ is a projective formula. The result follows by proceeding as in the proof of Theorem 4.1, but bearing in mind that there is an edge $(x,y)$ in the graph described by $\psi$ if and only if $\psi(x,y)$ or $\psi(y,x)$ holds (in some structure). \qed

It is unknown whether the version of CERTAIN 3-COLOURING OF BOOLEAN EDGE-LABELLED GRAPHS where the edges are labelled with Boolean literals, as opposed to disjunctions of Boolean literals, is also complete for $\Pi_2^p$ via projection translations.

We close this section by noting that our techniques developed in this paper require that problems be complete for complexity classes via projection translations. We cannot obtain Boolean labelled discrete structures by considering complete problems for complexity classes via quantifier-free translations, say: the restrictiveness of the projection translation is required.

5. A negative result.

Whilst problems have been shown to be complete for various complexity classes via projection translations ([Imm87], [IL89], [Ste91a], [Ste91b], [Ste91c]), it is unknown whether any problem is complete for any complexity class via monotone projection translations. However, Dahlhaus showed that an encoding of the SATISFIABILITY PROBLEM is a monotone quantifier-free translation of any problem in $\text{NP}$ which can be represented by a second-order sentence of a certain logical structure. We now show, by considering Boolean labelled digraphs, that it is unlikely that DTC($0, \text{max}$) (resp. TC($0, \text{max}$), HP($0, \text{max}$)) is complete for $\text{L}$ (resp. $\text{NL}$, $\text{NP}$) via monotone projection translations.
Theorem 5.1. If $DTC(0, \text{max})$ is complete for $L$ via monotone projection translations then $L = \text{NP}$.

Proof. Suppose that $DTC(0, \text{max})$ is complete for $L$ via monotone projection translations. Then every problem in $L$ can be represented by a sentence of the form:

$$DTC[\lambda xy \psi(x, y)](0, \text{max}),$$

where $\psi(x, y)$ is a monotone projective formula and $x$ and $y$ are $k$-tuples of distinct variables: in particular, $\psi(x, y)$ is of the form:

$$(\alpha_1 \land \beta_1) \lor (\alpha_2 \land \beta_2) \lor \ldots \lor (\alpha_m \land \beta_m),$$

where:

(i) each $\alpha_i$ is a conjunction of the logical atomic relations $s$, $=,$ and their negations;

(ii) each $\beta_i$ is atomic;

(iii) for $i \neq j$, $\alpha_i$ and $\alpha_j$ are mutually exclusive.

Hence, by [Fag74], every problem $\Omega$ of $\text{co-NP}$ can be represented by a second-order sentence of the form:

$$\forall R_1 \forall R_2 \ldots \forall R_t DTC[\lambda xy \psi(x, y)](0, \text{max}),$$

where each $R_i$ is a relation symbol of arity $a_i$. Assume that $\beta_1, \beta_2, \ldots, \beta_t$ involve some relation symbol from $\{R_1, R_2, \ldots, R_r\}$ while $\beta_{t+1}, \beta_{t+2}, \ldots, \beta_m$ do not.

Let $S \in \text{STRUCT}(\tau(\Omega))$ be of size $n$. Consider the Boolean edge-labelled digraph $G^S$ on the vertices $\{x : x \in |S|^k\}$ obtained as follows: for each $x, y \in |S|^k$:

(i) there is an edge $(x, y)$ in $G^S$ if and only if:

$$(S, x, y) \models \alpha_1 \lor \ldots \lor \alpha_t \lor (\alpha_{t+1} \land \beta_{t+1}) \lor \ldots \lor (\alpha_m \land \beta_m);$$

(ii) an edge $(x, y)$ of $G^S$ is labelled with the Boolean variable $X_j(z)$ if and only if:

$$(S, x, y) \models \alpha_j \text{ and the argument of } \beta_j(x, y) \text{ is } z, \text{ for some } j = 1, 2, \ldots, t;$$

(iii) an edge $(x, y)$ of $G^S$ is labelled with $\text{True}$ if and only if:

$$(S, x, y) \models \alpha_j \lor \beta_j, \text{ for some } j = t+1, t+2, \ldots, m.$$

Notice that $G^S$ is well-defined (as $\psi$ is a projective formula) and the labels are all positive (as $\psi$ is a monotone projective formula). It is easy to see that $S \in \Omega$ if and only if for all truth assignments $t$ on the Boolean variables involved as labels of edges of $G^S$, there is a path in $t(G^S)$ from vertex 0 to vertex $n-1$. Notice that the Boolean edge-labelled digraph $G^S$ can clearly be constructed from $S$ in logspace.

Let the decision problem PROB be defined as follows:

Instance of size $n$: a Boolean edge-labelled digraph $G$ on the vertices $\{0, 1, \ldots, n-1\}$ such that all labels of edges are Boolean variables from the set $\{X_{ij} : i, j = 0, 1, \ldots, n-1\}$;
Yes-instance of size n: an instance of size n such that for all truth assignments 
t on \{X_{i,j} : i,j = 0,1,...,n-1\}, there is a path in t(G) from
vertex 0 to vertex n-1. Then clearly, from above, PROB is complete for co-NP via logspace reductions.
Also, it is easy to see that PROB can be solved in logspace as all we need to do is
check that there is a path in G from vertex 0 to vertex n-1 such that each vertex
(except perhaps for vertex n-1) has out-degree 1 and is labelled True: otherwise,
there would exist some truth assignment t on the labels of the edges involved that
would disconnect vertex 0 and vertex n-1. Hence, co-NP = L, and so NP = L. □

Corollary 5.2. If TC(0,max) is complete for NL via monotone projection
translations then NL = NP.

Proof. Similar to that of Theorem 5.1. □

Corollary 5.3. If HP(0,max) is complete for NP via monotone projection
translations then NP = co-NP.

Proof. Similar to that of Theorem 5.1. □

6. Conclusion.

We feel that our technique of labelling discrete structures with Boolean literals
(or formulae) has more applications than those presented here: we intend to
consider the technique in more detail and on other discrete structures in future. It
should be apparent that showing problems to be complete for complexity classes
via extremely weak reductions, especially the logical reductions of this paper, is
important as not only can new complete problems be exhibited from old but we can
often derive "lower bound results" (as we have done in Theorem 5.1 and
Corollaries 5.2 and 5.3). We hope that the results of this paper encourage more
researchers to approach complexity theory from the point of view of finite model
theory.

References.

[AF90] M. AJTAI AND R. FAGIN, Reachability is harder for directed than
for undirected finite graphs, J. Symbolic Logic, 55, No. 1 (1990),
113-150.


