Annals of Discrete Mathematics 24 (1985) 51-72

Recurring Dominoes: Making the Highly Undecidable Highly Understandable

David Harel

Department of Applied Mathematics
The Weizmann Institute of Science
Rehovot, 76100 Israel

Abstract.

In recent years many diverse logical systems for reasoning about programs have been shown to posses a highly undecidable, viz Π_1^1-complete, validity problem. All such known results are reproved in this paper in a uniform and transparent manner by reductions from *recurring domino problems*. These are simple variants of the classical unbounded domino (or tiling) problems introduced by Wang and the bounded versions defined by Lewis. While the former are (weakly) undecidable and the latter complete in various complexity classes, the problems in the new class are Σ_1^1-complete.

It is hoped that the paper, which contains also NP-, PSPACE-, Π_1^0- and Π_2^0-hardness results for various logical systems, will enhance interest in the appealing medium of domino problems as a useful set of reduction tools for exhibiting "bad behavior".

1. Introduction

About two decades ago Hao Wang introduced *domino problems* [W1]. A *domino* is a 1×1 square tile, fixed in orientation, each of whose edges is associated with some color. In general, a domino problem is a decision problem that asks whether or not it is possible to tile some portion P of the integer grid $G = Z \times Z$ by dominoes of certain types, with perhaps some constraints on the placement of certain dominoes, colors or combinations thereof. The input to such a problem always includes some finite set $T = \{d_0, \ldots, d_m\}$ of domino types, consisting of the colors on the sides of each; one assumes the existence of an infinite supply of dominoes from each of the types in T. The general rules of tiling are that each grid point of P be associated with a single domino type from T, and that adjacent edges be monochromatic.

The problems introduced by Wang are characterized by the fact that the portion of G to be tiled is unbounded; it is either G itself, or a quadrant, halfgrid, octant, etc. The constraints in these *unbounded domino problems* are of finitary nature; some tile is required to appear, say, at the origin, the boundary of the quadrant (if that be the case) is to be colored, say, white; the dominoes occurring, say, along the diagonal are to be in some specified $T' \subseteq T$, etc. All these unbounded problems are undecidable but are co-r.e.. In other words, they are Π_1^0-complete; i.e., they and their complements reside at the base of the arithmetical hierarchy [R].

With the notable exception of the constraint-free versions (e.g. "can T tile G?") which are more difficult [Be], undecidability is established by setting up a straightforward correspondence between tiled rows of the portion P to be tiled and legal configurations of Turing machines (TM's). This is done such that adjacent rows correspond to legal transitions of the machine at hand, and the constraints are used to enforce an initial configuration containing the start state and, say, a blank tape. Given a TM M one can thus construct a set T_M of domino types that can tile P in accordance with a particular unbounded problem iff M halts on empty tape. Unbounded domino problems have been extensively investigated and have been used widely for proving undecidability of subcases of the decision problem for the predicate calculus (cf. [E, GK, G, KMW, L2, LP, W2]). The underlying idea in these proofs is pre-domino, and is rooted in work of Buchi [Bu].

Another class of domino problems, characterized by ***bounded*** portions P of the grid G, appears in work of Lewis [L1]. These ***bounded domino problems*** are complete in various complexity classes such as NP and PSPACE, and the hardness directions of these facts are established by similar reductions from TM computations. Bounded dominoes have been used in [L1, LP, E] for exhibiting intractability of certain problems, including the NP-completeness of satisfiability in the propositional calculus.

In [E] van Emde Boas presents strong arguments to the effect that the combinatorical and geometrical simplicity of domino problems renders them an ideal medium for introducing and proving "bad behavior" such as NP-hardness or undecidability. He suggests that they be used as "master reductions" in such lower-bound proofs. Indeed, one can easily describe domino problems to a novice, and unless a proof from first principles (i.e., computing machines such as TM's) is required, the proofs by reduction from dominoes are usually easy to present and to comprehend. Thus, domino problems can be regarded as an appealing abstraction capturing the two-dimensional time/space character of computation, but one which is devoid of the details of particular computing machines.

The existence of a third class of useful domino problems has been recently noticed in Harel [H2]. Problems in this class are obtained from unbounded domino problems simply by requiring that a designated domino, color or finitary combination thereof occur infinitely often in the tiling. These ***recurring domino problems*** (not to be confused with ***periodic*** tilings as in [GK, Be, Ro]) are shown in [H2] to be Σ_1^1-complete, i.e., to reside at the base of the highly undecidable analytical hierarchy [R]. The proof of these facts is similarly based on reductions from TM's but here the correspondence is with infinite computations of ***nondeterministic*** TM's (NTM's) which reenter a "signalling situation" infinitely often. As an illustration of the usefulness of recurring dominoes, an easy, almost trivial proof of the known Σ_1^1-hardness of satisfiability in either infinitary logic [K] or first-order (quantificational) dynamic logic [P, HMP] is presented in [H2]. This is actually an extension of the unbounded domino proof of undecidability of the predicate calculus, which in turn can be seen to be an extension of a bounded domino proof of NP-completeness for the propositional calculus (cf. [LP]).

The purpose of the present paper is to reinforce the arguments of [E, H2] by presenting weighty evidence of the usefulness of domino problems, in particular recurring domino problems, for bounding from below the complexity of decision problems in logical systems, especially program-oriented ones such as dynamic and temporal logics.

Many Π_1^1-completeness results for deciding validity in certain programming logics (i.e., Σ_1^1-completeness for satisfiability) have been established in the past few years using various techniques. These systems are quite diverse both in their motivation and applications and in their expressive power. Examples are quantificational dynamic logic, two-dimensional temporal logic, and context-free propositional dynamic logic.

The bulk of the present paper is Section 4 which is devoted to presenting proofs of all these Π_1^1-completeness results by reductions from recurring dominoes. The moral of this unifying exercise is that when dealing with logics of programs "when you have a grid you have it all"; that is, once one has forced candidate models of the formulas at hand to correspond to a manageable grid (usually the positive quadrant of G), the rest of a Π_1^1-hardness proof follows effortlessly by reduction from recurring dominoes. The grid-forcing part varies from trivial to quite tricky among the results presented, but in each case (i) the resulting proof is considerably easier and more transparent than the original one, and (ii) the grid-forcing part usually appears buried in the original proof anyway. Section 4 also contains domino proofs of various PSPACE- and Π_2^0-hardness results for logical systems, as well as some hitherto unpublished Π_1^1-results.

Section 2 defines the specific domino problems used in the sequel and states their complexity; it then provides some background on the logical systems discussed. Section 3 presents the three-part warm-up proof given in [H2] that the satisfiability/validity problem for the classical languages behaves as in the following table (in which notation of the polynomial-time, arithmetical, and analytical hierarchies has been used to emphasize uniformity):

formalism	satisfiability/validity	reduction from
propositional logic	Σ_1^p/Π_1^p	bounded dominoes
predicate logic	Π_1^0/Σ_1^0	unbounded dominoes
infinitary logic (constructive)	Σ_1^1/Π_1^1	recurring dominoes

It is hoped that the exposition presented herein will guide intuition as to the possible high undecidability of logical systems introduced in the future and will ease proofs of Π_1^1 results (and other lower bounds) by encouraging reductions from domino problems.

2. Preliminaries

2.1 Domino Problems and Their Complexity

The input to a domino problem always includes a finite set $T = \{ d_0, \ldots, d_m \}$ of domino types, each of the form $d_i = (left_i, right_i, up_i, down_i)$, giving the four colors associated with the sides of d_i. Colors are taken from some denumerable set C. Let $G = Z \times Z$, $G^+ = \mathbb{N} \times \mathbb{N}$, and $G^{++} = \{ (i,j) \mid (i,j) \in G^+, \text{ and } i < j \}$.

Following are some particular domino problems utilized in the sequel.

Bounded Problems:

B1: Given T and n (in unary), can T tile an $n \times n$ subgrid of G?

B2: Given T, n (in unary) and two colors c_0, c_1, can T tile some $n \times m$ subgrid of G such that the leftmost colors on the bottom and top are c_0 and c_1, respectively?

Unbounded Problems:

U1: Given T, can T tile G?

U2: Given T, can T tile G^+?

U*: Given T and two colors c_0, c_1, can T tile G^+ such that the sequence of colors on the bottom of the first row is of the form $c_0^n c_1^\omega$ for some n?

Recurring Problems:

R1: Given T; can T tile G such that d_0 occurs in the tiling infinitely often?

R2: Given T; can T tile G^+ such that d_0 occurs in the tiling infinitely often in the first column?

R3: Given T; can T tile G^{++} such that d_0 tiles at least one point in each row-column combination
$G_i = \{ (j,i) \mid 0 \leq j < i \} \cup \{ (i,j) \mid i < j \}$?

We assume familiarity with the hierarchy notation of Rogers [R], and with standard notions of complexity theory. In particular, we shall be using NP and PSPACE to stand for the set of problems decidable by a nondeterministic Turing machine in polynomial time and space, respectively. (By [Sa] the adjective "nondeterministic" is redundant for the latter.) NP is denoted Σ_1^p in the notation of the polynomial-time hierarchy [St], and PSPACE contains that entire hierarchy.

Σ_1^0 is the class of r.e. sets and its complement, Π_1^0, is the class of co-r.e. sets. Π_2^0 consists of all those sets (such as the codes of everywhere-halting TM's) which can be characterized by formulas over $\mathbb{N}$ of the form $\forall x \exists y R$ for recursive R; its complement is Σ_2^0. These classes reside low in the arithmetical hierarchy [R].

The class Σ_1^1 and its complement Π_1^1 reside low in the analytical hierarchy and represent sets characterizable, respectively, by formulas over $\mathbb{N}$ of the forms $\exists f R$ and $\forall f R$ for arithmetical R.

<u>Fact:</u>
(i) B1 is NP-complete, cf. [L1, LP, E].
(ii) B2 is PSPACE-complete, [L1, E].
(iii) U1 and U2 are Π_1^0-complete, [Be, Ro].
(iv) U* is Σ_2^0-complete, cf. [H2].
(v) R1, R2 and R3 are Σ_1^1-complete, [H2].

For details and discussions of these and other domino problems the reader is referred to van Emde Boas [E] and Harel [H2]. Problems U1 and R1 are not used in the paper and are described in order to exhibit the "cleanest" known unbounded and recurring versions.

2.2 The Logical Systems Considered

The systems considered in the sequel are of propositional and quantificational (first-order) character. We briefly describe each and provide references for details.

1) <u>Propositional calculus</u>, cf. [Me, Sh]:

Closure under $\vee$ and $\neg$ of propositional variables $P, Q, \ldots$ <u>Abbreviations:</u> $\wedge$, $\supset$, $\equiv$. Formulas satisfied by truth assignment to propositional variables.

2) <u>Propositional Temporal Logic of Linear Time</u> (TL), cf. [Pn, MP]:

Closure under $\vee, \neg, \Diamond$ and $\bigcirc$ of $P, Q, \ldots$. <u>Abbreviations:</u> as above, and $\Box = \neg \Diamond \neg$. Formulas satisfied in linear models of order-type ω in which each point supplies a truth assignment for the $P, Q, \ldots$ $\Diamond$ means "eventually", $\bigcirc$ means "at the next instant". <u>Example:</u> $P \wedge \Box \Diamond (Q \supset \bigcirc P)$, "$P$ true now and infinitely often Q implies P at the next instant".

3) <u>Two-Dimensional Temporal Logic</u> (2TL), cf. [HP]:

Closure under $\vee, \neg, \Diamond_i, \bigcirc_i$ for $i = 1, 2$, of $P, Q, \ldots$. <u>Abbreviations:</u> as above, and $\Box_i = \neg \Diamond_i \neg$. Formulas interpreted is cross-product of two TL models, i.e., in grids G^+, with $\Diamond_i, \bigcirc_i$ indicating progress along the two coordinates. <u>Example:</u> $\Box_1 \Diamond_2 P$, "P is true at least once on each column of the grid".

4) <u>Temporal-Spatial Logic</u> (TSL), cf. [RS]:

Closure under $\vee, \neg, \Diamond, \bigcirc$ <u>somewhere</u> and $\underline{L}$ of $P, Q, \ldots$. <u>Abbreviations:</u> as in TL, and <u>everywhere</u> = $\neg$<u>somewhere</u>$\neg$. Formulas interpreted in "proper interpretations" of networks of processors, with $\Diamond$ and $\bigcirc$ referring to time, $\underline{L}$ the (spatial) immediate connection between processors, and <u>somewhere</u> the reflexive transitive closure of $\underline{L}$. Proper interpretations have a grid-like structure just as in 2TL. <u>Example:</u> $P \wedge \Diamond \underline{L} (Q \vee \underline{\text{everywhere} P})$, "$P$ true now and here, and eventually the neighboring process satisfies Q or all its connected processes satisfy P".

5) Propositional Dynamic Logic (PDL), cf. [FL, H1]:

Formulas are closure under $\vee, \neg$, and $<\alpha>$ of $P, Q, \ldots$, where the programs α are regular expressions (i.e., closure under $\cup$, ;, *) of atomic programs $a, b, \ldots$ and tests $p?$ for formulas p. Abbreviations: as in prop. calc., and $[\alpha] = \neg<\alpha>\neg$. Formulas satisfied in structures $(W, \tau, \rho)$with $\tau(P) \subseteq W$ and $\rho(a) \subseteq W \times W$, interpreting propositional variables as true or false in states (= elements of W), and atomic programs as binary relations on states. Regular operators interpreted in standard relational calculus manner. $<\alpha>$ means "it is possible to execute α such that". Example: $P \supset [\alpha^*](R \vee <b \cup c^*d>Q)$, "if P is true then any terminating finite sequence of executions of a leads to a state in which either R is true or it is possible to execute either b or some number of c's followed by d and reach a state satisfying Q".

6) PDL With Additional Programs, cf. (HPS, HPa]:

Certain single nonregular programs, such as $L_1 = \{ a^i b a^i \mid i \geq 0 \}$, denoted $a^\Delta b a^\Delta$, or $L_2 = \{ a^{2^i} \mid i \geq 0 \}$ are added to PDL. Example: $[L_2][a]P$, "P is true at all points along a-paths at distance $2^i + 1$, for $i \in \omega$".

7) Deterministic PDL With Intersection (DPDL +"$\cap$"), cf. [HV]:

PDL interpreted in structures in which $\rho(a)$ is a function, and enriched with the intersection operator on programs; $\rho(\alpha \cap \beta) = \rho(\alpha) \cap \rho(\beta)$. Example: $<a \cap$ true?$>$ true, "there is an effectless execution of a".

8) Inference and Implication in PDL cf. [MSM]:

A formula of PDL containing a "free" propositional variable Q, and denoted $A(Q)$ is regarded as an axiom scheme, standing for the set A(PDL) of all formulas obtained from it by consistently substituting arbitrary PDL formulas for Q. A(Q) infers P if P is valid in all structures in which all formulas of A(PDL) are. A(Q) implies P if P is true in any state in which all formulas of A(PDL) are. Example: $<a>Q \supset [a]Q$ infers all formulas true for deterministic a.

9) Global Process Logic, cf. [HKP, S]:

Closure under $\vee$, $\neg$, $<\alpha>$, first and suf of $P, Q, \ldots$. Programs are as in PDL. Abbreviations: as in PDL. Formulas satisfied by paths in structures (W, τ, ρ), where $\tau(P) \subseteq W^*$ and $\rho(a) \subseteq W^*$, interpreting both formulas and programs as paths of states. $<\alpha>P$ satisfied in path p if P is satisfied in some path pq with $q \in \rho(\alpha)$. The connective first, for example, is unary and first$P$ is satisfied in p if P is satisfied in the first state of p. A derived operator next is defined so that p satisfies next$P$ if the greatest proper suffix of p satisfies P. For each i a formula L_i can be defined, true in all paths of length i. The operator last is also derived. Example: $L_0 \wedge <\alpha^*>($last$[b]$next$Q))$, is true in "paths consisting of a single state, in which there is some a^* path, at the end of which each b path is such that the path obtained by truncating the first state satisfies Q".

10) Predicate Calculus (with equality), cf. [Me, Sh]:

Closure under $\vee$, $\neg$, $\exists x$ of atomic formulas $P(t_1, \ldots, t_n)$ for terms t_i. Language includes binary predicate "=". Formulas interpreted in first-order structures, "=" interpreted as equality. Abbreviations: as in prop. calc., and $\forall x = \neg \exists x \neg$.

11) Augmented Arithmetic, cf. [Sh]:

Predicate calculus with $=$, 0, 1, $+$, x, $<$ interpreted over $\mathbb{N}$ in the standard way, augmented with extra uninterpreted predicate symbols.

12) Infinitary Logic (constructive version is denoted $L^{CK}_{\omega_1\omega}$), cf. [K]:

Predicate calculus closed also under ω-disjunctions and conjunctions. In the constructive version disjunctions and conjunctions are r.e. Example: $\bigvee_{i\in\omega}\varphi_i$, where $\varphi_0 = \neg \exists x(x = x)$, $\varphi_1 = \exists x \forall y(x = y)$, $\varphi_2 = \exists x \exists y(\neg x = y \wedge \forall z(x = z \vee y = z))$, etc., is true precisely in finite structures.

13) Quantified Temporal Logic of Linear Time (QTL), cf. [Pn]:

Closure under $\vee$, $\neg$, $\diamond$, $\bigcirc$ and $\exists x$ of atomic formulas as in pred. calc. Abbreviations: as in TL and pred. calc. Formulas satisfied in linear models of order-type ω underlying first-order structures. Each point in time provides values for all variables. Example: $x = y \supset \diamond \square(P(x) \wedge x = y)$, "if $x = y$ then from some future point on $x = y$ and P is true of x".

14) Quantified Dynamic Logic (QDL), cf. [P, H1]:

Closure under $\vee$, $\neg$, $<\alpha>$, $\exists x$ of atomic formulas, where the programs α are regular expressions over assignments $x \leftarrow t$ for term t and tests $\varphi?$ for formula φ. Abbreviations: as in PDL and pred. calc. Formulas satisfied in first-order structures in which states provide values for variables; assignments interpreted in the standard way, and program operations as in PDL. Example: $x = y \supset [(x \leftarrow f(f(x)))^*] < (y \leftarrow f(y))^* > x = y$, is valid states "every execution of $(ff)^*$ corresponds to some execution of f^*".

15) Repeating in QDL, cf. [H1]:

The predicate repeat(α) for α a QDL program states that α can be repeated indefinitely. Formulas satisfied in QDL structures. Example: repeat$((x \leftarrow f(x); P(x)?)$, states "for all i, $P(f^i(x))$ is true".

We remark that the rest of the paper presents only the hardness directions of the results. All results are actually completeness results and the upper bounds are usually easy to establish. In particular, Σ^1_1-hard satisfiability problems can be shown to be in Σ^1_1 by appealing to an appropriate version of the Löwenheim-Skolem Theorem, and writing "φ is satisfiable" as "$\exists$ countable structure ...", which is Σ^1_1.

3. Classical Systems

In all reductions we assume an input set $T = \{ d_0, \ldots, d_m \}$ involving colors $C_T = \{ c_0, \ldots, c_{k-1} \}$, where w.l.o.g. k is a power of 2. We use propositional or predicate symbols denoted by LEFT, RIGHT, UP, DOWN, indexed by superscripts $1 \leq u \leq \log k$. In propositional logics we might have additional subscripts yielding, for example, $\mathrm{LEFT}^u_{i,j}$, or LEFT^u_i, for $1 \leq i,\ j \leq n$, and in quantificational (= first-order) logics the predicates are binary or unary, as in $\mathrm{LEFT}^u(x,y)$ or $\mathrm{LEFT}^u(x)$. In general, unsuperscripted symbols stand for the appropriately ordered sets of the $\log k$ superscripted ones; e.g., LEFT_i is $(\mathrm{LEFT}^1_i, \ldots, \mathrm{LEFT}^{\log k}_i)$. In this way, indentifying color c_i with the binary representation of i, the colors in C_T are in a fixed one-to-one correspondence with possible truth assignments to such sets. Accordingly, we shall write, say, $\mathrm{RIGHT}_{i,j} = c_\ell$ as an abbreviation of the appropriate conjunction of the $\pm\mathrm{RIGHT}^u_{i,j}$, to indicate that the color on the right hand edge of the domino associated with (i,j) is c_ℓ or $\mathrm{UP}(x,y) = \underline{\mathrm{down}}_n$ to mean that the color on the top edge of the domino at (x,y) is that on the bottom of domino d_n in T. To associate dominoes from T with such indices we write, e.g., $\mathrm{LRUD}_{i,j} = d_n$ as an abbreviation of $\mathrm{LEFT}_{i,j} = \underline{\mathrm{left}}_n \wedge \mathrm{RIGHT}_{i,j} = \underline{\mathrm{right}}_n \wedge \mathrm{UP}_{i,j} = \underline{\mathrm{up}}_n \wedge \mathrm{DOWN}_{i,j} = \underline{\mathrm{down}}_n$. Similarly for $\mathrm{LRUD}(x,y)$.

Theorem 3.1 [C]: Satisfiability in the propositional calculus is NP-hard.

Proof [LP]: Given T and n, construct $P_{T,n}$ as the conjunction of

$$\bigwedge_{i=1}^{n} \bigwedge_{j=1}^{n} \Big(\bigvee_{\ell=0}^{n} \mathrm{LRUD}_{i,j} = d_\ell \Big), \tag{1}$$

and

$$\bigwedge_{i=1}^{n-1} \bigwedge_{j=1}^{n} (\mathrm{RIGHT}_{i,j} = \mathrm{LEFT}_{i+1,j} \wedge \mathrm{UP}_{j,i} = \mathrm{DOWN}_{j,i+1}). \tag{2}$$

Clearly $P_{T,n}$ is of size a polynomial in $n+m$, and is satisfiable iff T,n satisfies B1. The latter is seen by observing that (1) associates a domino from T with each point of $[1\ldots n] \times [1\ldots n]$, and (2) asserts correct matching of colors. ∎

Theorem 3.2 [Ch,T]: Satisfiability in the predicate calculus is Π^0_1-hard.

Proof: Given T, construct φ_T as the conjunction of

$$\forall x (f(x) \neq z \wedge \forall y (f(x) = f(y) \supset x = y)), \tag{0}$$

$$\forall x \forall y \Big(\bigvee_{\ell=0}^{m} \mathrm{LRUD}(x,y) = d_\ell \Big), \tag{1}$$

and

$$\forall x \forall y(\mathrm{RIGHT}(x,y) = \mathrm{LEFT}(f(x),y) \wedge \mathrm{UP}(x,y) = \mathrm{DOWN}(x,f(y))). \tag{2}$$

The claim is that φ_T is satisfiable iff T satisfies U2. The if direction is trivial since if a tiling exists φ_T is satisfied in $\mathbb{N}$ with z interpreted as 0 and f as successor. Conversely, the domain of any structure satisfying φ_T must contain, by clause (0), an infinite set S constituting the values of $z, f(z), f(f(z)), \ldots$. The grid G^+ matches $S \times S$, with (i,j) corresponding to $(f^i(z), f^j(z))$. Clause (1) and (2) behave as in Thm. 3.1, yielding a tiling of G^+. ∎

Theorem 3.3 (cf. [K,R]): Satisfiability in constructive infinitary logic is Σ_1^1-hard.

Proof [H2]: Given T construct φ'_T as the conjunction of φ_T of Thm. 3.2, and

$$\forall x \bigvee_{i\in\omega}(\mathrm{LRUD}(z, f^i(x)) = d_0). \tag{3}$$

The claim is that φ'_T is satisfiable iff T satisfies R2. The if direction is as before but now (3) holds by virtue of the recurrence of domino d_0. Conversely, if (3) holds, d_0 occurs arbitrarily high up in the first column $(z, \{f^i(z)\}_{i\in\omega})$ of G^+. ∎

Theorem 3.4 (cf. [R]): Satisfiability in augmented first-order arithmetic is Σ_1^1-hard.

Proof: There is no need for clause (0) of Thm. 3.2. Given T, construct ψ_T as the conjunction of

$$\forall x \forall y(\bigvee_{\ell=0}^{m} \mathrm{LRUD}(x,y) = d_\ell), \tag{1}$$

$$\forall x \forall y(\mathrm{RIGHT}(x,y) = \mathrm{LEFT}(x+1,y) \wedge \mathrm{UP}(x,y) = \mathrm{DOWN}(x,y+1)), \tag{2}$$

and

$$\forall x \exists y(x < y \wedge \mathrm{LRUD}(z,y) = d_0). \tag{3}$$

ψ_T is satisfiable iff T satisfies R2. ∎

4. Dominoe Proofs of Hardness Results in Logics of Programs

The role played by the four conjuncts (0)–(3) in the proofs in the previous section is the connecting thread of all proofs in this one.

While clause (0), which states that "models look like grids", will take on quite diverse forms in the sequel, the general form of (1), (2) and (3) will be

1. $\forall$point (point $\in T$)
2. $\forall$point (colors match right and above neighbors)
3. $\forall$distance $\exists$point-further-away (point is d_0).

Since in most cases the parenthesized parts are written easily using the abbreviations introduced in Section 3, one really has to specify only how to universally reach all points on the appropriate grid, how to existentially reach points "further away" than the "current" one, and how to reach the neighbors of any "current" point.

We first discuss the quantificational logics QDL and QTL.

Theorem 4.1 ([M], cf. [HMP]): Satisfiability in QDL is Π^1_1-hard, even for formulas of the form $\forall x \langle (x \leftarrow f(x))^* \rangle \varphi$ for program-free φ.

Proof: Replace (3) in the proof of Thm. 3.3 by

$$\forall x \langle (x \leftarrow f(x))^* \rangle (\mathrm{LRUD}(z, x) = d_0). \tag{3}$$

Since the program within the $\langle\rangle$ does not involve z (= the only free variable in $(0) \wedge (1) \wedge (2)$), the final formula ψ_T can be taken to be $\forall x \langle (x \leftarrow f(x))^* \rangle (\mathrm{LRUD}(z, x) = d_0 \wedge \varphi_T)$, where φ_T is as in the proof of Thm. 3.2. It is satisfiable iff T satisfies R2. ∎

Theorem 4.2 [H1]: Satisfiability of formulas of the form repeat(α), for QDL programs α with program-free tests, is Σ^1_1-hard.

Proof: Given T, let α_T be

$$\varphi_T?; \quad (x \leftarrow f(x))^*; \quad x \leftarrow f(x); \quad (\mathrm{LRUD}(z, x) = d_0)?$$

where φ_T is as in the proof of Thm. 3.2. An indefinite repetition of α_T is possible only if φ_T is satisfied in a state that admits an infinite computation of $x \leftarrow f(x)$, with d_0 recurring along the first column of the grid G^+. Hence repeat(α_T) is satisfiable iff T satisfies R2. ∎

In contrast to Thm. 4.2, satisfiability of formulas of the form loop(α) for QDL programs α with program-free tests can be shown to Π^0_1-compete. See [H1].

Theorem 4.3: Satisfiability in QTL is Π_1^1-hard, even for formulas of the form $\Box\Diamond\varphi$ for φ involving only a single $\bigcirc$ (and no $\Box$ or $\Diamond$).

Proof: Given T, construct ψ_T as the conjunction $\Box\varphi$, where φ is (0) of the proof of Thm. 3.2, and

$$\Box\forall x\Big(\bigvee_{\ell=0}^{m} \mathrm{LRUD}(x) = d_\ell\Big), \tag{1}$$

$$\Box\forall x(\mathrm{RIGHT}(x) = \mathrm{LEFT}(f(x)) \wedge \bigwedge_{i=0}^{k-1}(\mathrm{UP}(x) = c_i \supset \bigcirc(\mathrm{DOWN}(x) = c_i))), \tag{2}$$

$$\Box\Diamond(\mathrm{LRUD}(x) = d_0). \tag{3}$$

Here the infinite set z, $f(z)$, $f(f(z)), \ldots$ of clause (0) is used only in the horizontal direction; the vertical one is modeled by the temporal axis. The special form is obtained as in Thm. 4.1 and is satisfiable iff T satisfies R2. ∎

It is possible to prove the Π_2^0-hardness of validity of Hoare [H]partial correctness assertions using the U* domino problem. To see this note that $\psi\{\alpha\}\varphi$ is $\psi\supset[\alpha]\varphi$ in QDL notation, or $[\psi?;\alpha]\varphi$. The following Theorem thus gives the result.

Theorem 4.4 [HMP]: Satisfiability of formulas of QDL of the form $<\alpha>\varphi$ for program-free φ is Σ_2^0-hard, even for test-free α.

Proof: Given T, construct ψ'_T to be the conjunction of φ_T from Thm. 3.2, and

$$\begin{aligned}\mathrm{LRUD}(z,z) = d_0 \wedge \forall x((&\mathrm{DOWN}(x,z) = c_0 \supset \\ &(\mathrm{DOWN}(f(x),z) = c_0 \vee \mathrm{DOWN}(f(x),z) = c_1)) \wedge \\ &(\mathrm{DOWN}(x,z) = c_1 \supset \mathrm{DOWN}(f(x),z) = c_1)).\end{aligned} \tag{*}$$

Now let ψ_T be

$$<x \leftarrow z; (x \leftarrow f(x))^*>(\mathrm{DOWN}(x,z) = c_0 \wedge \mathrm{DOWN}(f(x),z) = c_1 \wedge \psi'_T).$$

Given that $\underline{\mathrm{down}}_0 = c_0$, ψ'_T forces the bottom colors on the bottom row of G^+ to be either c_0^ω or $c_0^n c_1^\omega$ for some n. ψ_T then prevents the first possibility and hence is satisfiable iff T satisfies U*. ∎

Turning now to the propositional logics PDL, TL and PL, the results here are at times somewhat more involved due to the difficulty of forcing models to look like grids.

Theorem 4.5 [HP]: Satisfiability in 2TL is Σ_1^1-hard.

Proof: Given T, construct P_T as the conjunction of

$$\Box_1 \Box_2\Big(\bigvee_{\ell=0}^{m} \text{LRUD} = d_\ell\Big), \tag{1}$$

$$\Box_1 \Box_2\Big(\bigwedge_{i=0}^{k-1}((\text{RIGHT} = c_i \supset \underset{1}{\bigcirc}(\text{LEFT} = c_i)) \wedge (\text{UP} = c_i \supset \underset{2}{\bigcirc}(\text{DOWN} = c_i)))\Big), \tag{2}$$

and

$$\Box_2 \Diamond_2(\text{LRUD} = d_0). \tag{3}$$

P_T is satisfiable iff T satisfies R2. ∎

Theorem 4.6 [RS]: Satisfiability in TSL is Σ_1^1-hard.

Proof: Given T, construct P_T as the conjunction of

$$\Box\underline{\text{everywhere}}\Big(\bigvee_{\ell=0}^{m} \text{LRUD} = d_\ell\Big), \tag{1}$$

$$\Box\underline{\text{everywhere}}\Big(\bigwedge_{i=0}^{k-1}((\text{RIGHT} = c_i \supset \bigcirc(\text{LEFT} = c_i)) \wedge (\text{UP} = c_i \supset \underline{L}(\text{DOWN} = c_i)))\Big), \tag{2}$$

$$\underline{\text{everywhere}}(\underline{\text{somewhere}}(\text{LRUD} = d_0)). \tag{3}$$

P_T is satisfiable iff T satisfies R2. ∎

It is possible to prove that one-dimensional TL is PSPACE-hard using bounded dominoes (without $\bigcirc$ it is NP-complete [SC]):

Theorem 4.7 [SC]: Satisfiability in TL is PSPACE-hard.

Proof: Given T, n and colors c_0, c_1, construct P_{T,n,c_0,c_1} as the conjunction of

$$\Box(Q \supset \bigwedge_{i=1}^{n} \bigvee_{\ell=0}^{m} (\text{LRUD}_i = d_\ell)), \tag{1}$$

$$\Box\left(Q\supset\left(\bigwedge_{i=1}^{n-1}(\mathrm{RIGHT}_i=\mathrm{LEFT}_{i+1})\wedge\bigwedge_{i=1}^{n}\bigwedge_{j=0}^{k-1}(\mathrm{UP}_i=c_j\supset\bigcirc(Q\supset(\mathrm{DOWN}_i=c_j)))\right)\right), \tag{2}$$

$$Q\wedge\Box(\neg Q\supset\bigcirc\neg Q)\wedge\mathrm{DOWN}_1=c_0\wedge\Diamond(Q\wedge\bigcirc(\neg Q)\wedge\mathrm{UP}_1=c_1) \tag{*}$$

Process along the vertical $[1\ldots m]$ axis of B2 is achieved with the temporal operators, and the horizontal axis is bounded by n and referred to by $1\leq i\leq n$. P_{T,n,c_0,c_1} is thus satisfiable iff (T,n,c_0,c_1) satisfies B2, since (*) states that the first color on the first row matches c_0 and on some (further) one matches c_1. Throughout, Q is forced to be true precisely at the first m vertical points. ∎

It is possible to use the trick from [MS] combined with a reduction from B2, to obtain similar transparent domino proofs of PSPACE-hardness for quantified Boolean formulas [MS], the first-order theory of equality [MS], and certain systems of modal logic [La]. We omit the details here.

<u>Theorem 4.8</u> [MSM]: The non-inference and non-implication problems for PDL are Σ_1^1-hard.

<u>Proof:</u> Let $A(Q)$ be

$$(<ab>Q\supset[ba]Q)\wedge(<ba>Q\supset[ab]Q).$$

Given T, construct P_T as the conjunction of

$$[(a\cup b)^*](<a>\underline{\mathrm{true}}\wedge<b>\underline{\mathrm{true}}), \tag{0}$$

$$[(a\cup b)^*]\left(\bigvee_{\ell=0}^{m}\mathrm{LRUD}=d_\ell\right), \tag{1}$$

$$[(a\cup b)^*]\left(\bigwedge_{i=0}^{k-1}((\mathrm{RIGHT}=c_i\supset[a](\mathrm{LEFT}=c_i))\wedge(\mathrm{UP}=c_i\supset[b](\mathrm{DOWN}=c_i)))\right), \tag{2}$$

and

$$[b^*]<b^*>(\mathrm{LRUD}=d_0). \tag{3}$$

Clause (0) forces the existence of a binary a,b tree from any satisfying state. The axiom scheme $A(Q)$, when regarded as the infinite set A(PDL), forces this tree to act, as far as can be detected by PDL formulas, like a grid. Specifically the claims are:

(i) A(PDL) infers $\neg P_T$ iff T does not satisfy R2;

(ii) $[(a\cup b)^*]$A(PDL) implies $\neg P_T$ iff T does not satisfy R2.

To see (i), if T satisfies R2 the structure consisting of a quadrant as in Fig. 1, tiled accordingly, satisfies A(PDL) in all states, but satisfies P_T at state s. Conversely, if all states of some structure satisfy A(PDL), and some state s satisfies P_T then the "forward part" from s [MSM] looks essentially like Fig. 1, by (0) and A(PDL). Clauses (1)–(3) then assert the existence of the required tiling. ∎

Remark: Our proof of Theorem 4.8 involves only test-free programs (cf. [MSM, Thm. 4.4]) and can be strengthened as in [MSM, Thms. 4.5, 4.6] to atomic-test-DPDL.

A very similar-looking proof can be given for deterministic PDL with intersection:

Theorem 4.9 [HV]: Satisfiability in DPDL + "$\cap$" is Σ_1^1-hard.

Proof: [HV]: Construct P_T as the conjunction of (1)–(3) of the previous proof, and,

$$[(a \cup b)^*](<ab \cap ba>\underline{\text{true}}). \tag{0}$$

Clause (0) forces the existence of a (possibly cyclic) grid. P_T is thus satisfiable iff T satisfies R2. ∎

Theorem 4.10 [HPS]: Satisfiability in PDL + $\{a^\Delta b a^\Delta\}$ is Σ_1^1-hard.

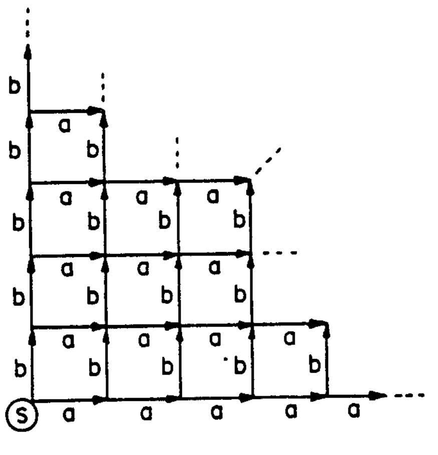

Figure 1.

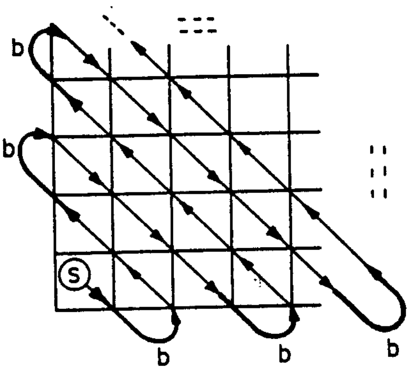

Figure 2.

Proof: Given T, denoting $a^{\Delta}ba^{\Delta}$ by L and a^*b by N, construct P_T as the conjunction of

$$<ab>\underline{\text{true}} \wedge [N^*](<a^*ab>\underline{\text{true}} \wedge [a^*a][L][ab]\underline{\text{false}} \wedge [L][aa]\underline{\text{false}}), \tag{0}$$

$$[(a \cup b)^*a]\Big(\bigvee_{\ell=0}^{m} \text{LRUD} = d_\ell\Big), \tag{1}$$

$$\begin{aligned}
&[(NN)^*a^*]\Big(\bigwedge_{i=0}^{k-1}((\text{RIGHT} = c_i \supset [L][a](\text{LEFT} = c_i)) \\
&\qquad\qquad \wedge (\text{UP} = c_i \supset [L][aa](\text{DOWN} = c_i)))\Big) \\
&\wedge[(NN)^*Na^*]\Big(\bigwedge_{i=0}^{k-1}((\text{RIGHT} = c_i \supset [L][aa](\text{LEFT} = c_i)) \\
&\qquad\qquad \wedge (\text{UP} = c_i \supset [L][a](\text{DOWN} = c_i)))\Big),
\end{aligned} \tag{2}$$

$$[(NN)^*]<(NN)^*>(\text{LRUD} = d_0). \tag{3}$$

Clause (0) forces the existence, in any potential model, of an infinite sequence of the form $\sigma = aba^2ba^3b\ldots$. Clause (1) associates dominoes from T with those points of σ that follow a's, and (2) forces the matching of colors, so that σ corresponds to G^+, as illustrated in Fig. 2. Note how neighbors from the right and from above are reached using L. Consequently, P_T is satisfiable iff T satisfies R2. ∎

Remark: As in the proof in [HPS] it is possible to modify this proof slightly and obtain the result for PDL + $\{a^{\Delta}b^{\Delta}, b^{\Delta}a^{\Delta}\}$. The question of whether PDL + $\{a^{\Delta}b^{\Delta}\}$ is decidable or not is still open, cf. [H1].

Theorem 4.11 [HPa]: Satisfiability in PDL + $\{L\}$, where $L = \{a^{2^i} \mid i \geq 0\}$, is Σ_1^1-hard.

Sketch of Proof [HPa]: Given T, construct P_T as the conjunction of the following formulas, which involve the additional predicate symbols Q_i, R_j, for $0 \leq i \leq 6$, $0 \leq j \leq 3$.

$$
\begin{aligned}
&[a^*]\Big(<a>\underline{\text{true}} \wedge \bigwedge_{0\le i<j\le 6} \neg(Q_i \wedge Q_j) \wedge \bigwedge_{0\le i<j\le 3} \neg(R_i \wedge R_j)\Big)\\
&\quad \wedge Q_0 \wedge [a^*]\Big(\bigwedge_{i=0}^{6} (Q_i \supset [a] Q_{(i+1)(\mathrm{mod}\ 7)})\Big) \qquad (0)\\
&\quad \wedge [L]R_3 \wedge [La]((Q_3 \supset R_1) \wedge (Q_5 \supset R_2))\\
&\quad \wedge [LL]((R_1 \supset [L](R_0 \vee R_2 \vee R_3)) \wedge (R_2 \supset [L](R_0 \vee R_1)) \wedge (R_0 \supset [L](R_1 \vee R_2))),
\end{aligned}
$$

$$
[LL]\Big(\neg R_3 \supset \bigvee_{\ell=0}^{m} \mathrm{LRUD} = d_\ell \Big), \qquad (1)
$$

$$
\begin{aligned}
[LL]\Big(\bigwedge_{i=0}^{2} \bigwedge_{j=0}^{k-1} (R_i \supset ((\mathrm{RIGHT} = c_j \supset [L](R_{(i-1)(\mathrm{mod}\ 3)} \supset \mathrm{LEFT} = c_j))\\
\wedge (\mathrm{UP} = c_j \supset [L](R_{(i+1)(\mathrm{mod}\ 3)} \supset \mathrm{DOWN} = c_j))))\Big),
\end{aligned} \qquad (2)
$$

and

$$
[L]<L>(\neg R_3 \wedge \mathrm{LRUD} = d_0). \qquad (3)
$$

Here the claim is that P_T is satisfiable iff T satisfies R3. This is rather difficult to see immediately, and the details of the proof appear in [HPa]. However, to get a feeling for it, clause (0) forces points at distances in $\{ 2^i + 2^j \mid i, j \geq 0 \}$ to form an octant grid as in Fig. 3. There, a parenthesized number is the subscript of that R_i forced to be true at the point. One sees that the element to the right of a point s satisfying R_i for $i \neq 3$, satisfies $R_{(i-1)(\mathrm{mod}\ 3)}$ and the one above it $R_{(i+1)(\mathrm{mod}\ 3)}$. Moreover, these are the only two points in G^{++} at distances a power of 2 from s. Thus, from any point on the superdiagonal portion G^{++} of this L^2 grid, any execution of L leads either to its neighbors or outside the grid. Clause (3) can be seen to state the recurrence property of R3, and for it to work it is essential that d_0 occurs in all G_i as in the statement of R3. ∎

Remark: It is open whether, e.g., PDL + L, for $L = \{ a^{i^2} \mid i \geq 0 \}$ or $L = \{ a^{i^3} \mid i \geq 0 \}$, is undecidable.

Theorem 4.12 [S]: Satisfiability in Global Process Logic is Σ^1_1-hard.

<u>Proof</u> (cf. [S]): Given T, construct P_T as the conjunction of

$$L_0 \wedge [a^*]\underline{\text{last}}([a]L_1), \tag{0}$$

$$[a^*]\underline{\text{last}}([a^*]\Big(\bigvee_{\ell=0}^{m} \text{LRUD} = d_\ell\Big), \tag{1}$$

$$[a^*]\underline{\text{last}}([a^*]\Big(\bigwedge_{i=0}^{k-1} ((\text{RIGHT} = c_i \supset [a](\text{LEFT} = c_i)) \wedge \\ (\text{UP} = c_i \supset [a]\underline{\text{next}}(\text{DOWN} = c_i)))\Big)), \tag{2}$$

$$[a^*]\underline{\text{last}}(<a^*a>\underline{\text{last}}(\text{LRUD} = d_0)). \tag{3}$$

Here the claim is that P_T is satisfiable iff T satisfies R2. Clause (0) together with the $<>$ part of (3), forces the existence of an implicit quadrant grid G^+ in which point (i,j) corresponds to the segment $P_{i,j} = (s_i, \ldots, s_{i+j})$ of an infinite path $p = (s_0, s_1, \ldots)$ with $(s_i, s_{i+1}) \in \rho(a)$ for each i. In this way, the right neighbor of $p_{i,j}$ is obtained by an execution of a, and the above neighbor by an execution of a followed by <u>next</u>. Executions of a^* followed by <u>last</u> correspond to arbitrary movement up a column; in this way (3) really asserts the recurrence property of R2. See Fig. 4. ∎

⋮		⋮		⋮	
33 (2)	34 (1)	36 (0)	40 (2)	48 (1)	64 (3)
17 (1)	18 (0)	20 (2)	24 (1)	32 (3)	
9 (0)	10 (2)	12 (1)	16 (3)		
5 (2)	6 (1)	8 (3)			
3 (1)	4 (3)				
2 (3)					

Figure 3.

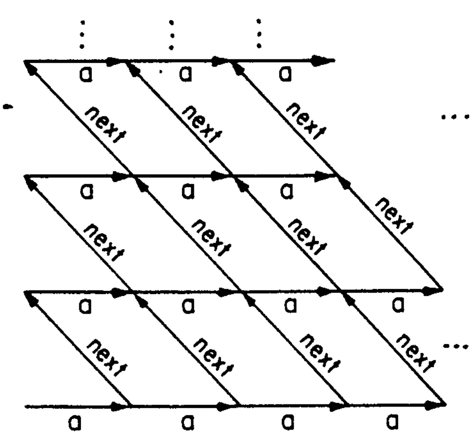

Figure 4.

5. Conclusions and Discussions

It is hoped that the simplicity and similarity of the proofs in Section 3 and 4 speak for themselves. All lower bounds of NP, PSPACE, Π_1^0, Π_2^0 and Σ_1^1 for satisfiability in logical systems which are known to the author have been provided with such proofs. It seems that domino problems, with their "$\exists$ tiling" format, are a perfect match for satisfiability problems, with their "$\exists$ model" format. It is the third class of domino problems, the recurring ones of [H2], that enable completion of this picture for the various programming logics.

Three general additional points are worth making.

1. Since all domino problems owe their complexity to the correspondence with Turing machine computations, and since this correspondence applies to nondeterministic models just as well ("$\exists$ tiling" corresponds to "$\exists$ computation"), cf. [H2], domino problems can apparently not distinguish between deterministic and nondeterministic classes. Thus, e.g., EXPTIME-hard satisfiability problems, such as that for PDL [FL], do not admit domino proofs, whereas the above mentioned classes all do (PSPACE does by Savitch's Theorem [Sa]).

2. Domino problems are existential in nature and do not seem to extend in any natural way to capture alternation. One additional quantifier can usually be managed, cf. the (relatively cumbersome) formulation of the Σ_2^0 problem U* used in the proof of Theorem 4.4. Thus, while games are good for alternation, dominoes are good for single existentials. Indeed the EXPTIME-hardness of PDL is proved using EXPTIME = alternating-linear-space, with alternating TM's, and as noted above cannot be proved using the kinds of domino problems considered herein.

3. The present paper and its companion [H2] make the case for viewing Σ_1^1 sets as corresponding to computable finitely-branching trees with an infinite path containing a recurrence. Call these F-trees. It is a well known fact that Σ_1^1 sets correspond to computable possibly infinitely-branching trees containing some infinite path. Call these I-trees. For example, the set of (notations for) recursive ordinals, of well-founded recursive trees, and of terminating computations of programs with unbounded nondeterminism, etc. are all Π_1^1-complete (cf. [R, Cn, AP]).

   The correspondence between these views can easily be visualized by traversing a computable infinitely-branching tree with an NTM which at each stage nondeterministically chooses to either move across to a brother or down to a son, signalling when the latter is chosen. Its computation tree is an F-tree with recurring signal iff the initial tree is an I-tree. Conversely, given a computable finitely branching tree with a "signal", an infinitely-branching tree can be constructed with nodes corresponding to signal nodes in the former, and a node's sons corresponding to all possible signalled descendents of the origin node. In particular, the recursive ordinal corresponding to a nonrecurring domino set $T = \{d_0, \ldots, d_m\}$ is that associated with the following tree: The root is associated with the 1×1 tiling consisting of d_0. The sons of each node are all possible minimal $n \times n$ extensions of the tiling associated with that node, for any possible n, which contain additional occurrences of d_0. This tree is well-founded iff T does not satisfy R1. (Similar constructions clearly exist for other recurring domino problems.)

This observation concerning "fat" and "thin" infinite trees is formalized in [H2], and the Σ_1^1-hardness of the recurring NTM's (from which recurring dominoes are derived) is obtained as a corollary. A significantly stronger correspondence result for infinite trees, which has applications to fair computations as well as to richer cases of the domino problem, will be published separately.

6. References

[AP] Apt, K.R. and G.D. Plotkin, Countable Nondeterminism and Random Assignment, Manuscript, 1982.

[Be] Berger, R., The Undecidability of the Dominoe Problem, *Mem. Amer. Math. Soc.* **66** (1966).

[Bn] Buchi, J.R., Turing Machines and the Entscheidungsproblem, *Math. Ann.* **148** (1962), 201–213.

[Ch] Church, A., A Note on the Entscheidungsproblem, *J. Symb. Logic* **1** (1936), 101–102.

[Cn] Chandra, A.K., Computable Nondeterministic Functions, *19th IEEE Symp. Found. comput. Sci.*, 127–131, 1978.

[C] Cook, S.A., The Complexity of Theorem Proving Procedures, *3rd ACM Symp. Theory of Comput.*, 151–158, 1971.

[E] van Emde Boas, P., Dominoes are Forever, *1st GTI Workshop*, Paderborn, 75–95, 1983.

[FL] Fischer, M.J. and R.E. Ladner, Propositional Dynamic Logic of Regular Programs, *J. Comput. Syst. Sci.* **18** (1979), 194–211.

[G] Gurevich, Y., The Decision Problem for Standard Classes, *J. Symb. Logic* **41** (1976), 460–464.

[GK] Gurevich, Y. and I.O. Koryakov, Remarks on Berger's Paper on the Domino Problem, *Siberian Math. J.* **13** (1972), 319–321.

[H1] Harel, D., Dynamic Logic, In: *Handbook of Philosophical Logic* II, Reidel (1984), to appear.

[H2] Harel, D., A Simple Highly Undecidable Domino Problem (or, A Lemma on Infinite Trees, With Applications), *Proc. Logic and Computation Conference.* Clayton, Victoria, Australia, Jan. 1984.

[HKP] Harel, D., D. Kozen and R. Parikh, Process Logic: Expressiveness, Decidability, Completeness, *J. Comput. Syst. Sci.* **25** (1982), 144–170.

[HMP] Harel, D., A.R. Meyer and V.R. Pratt, Computability and Completeness in Logics of Programs, *9th ACM Symp. Theory of Comput.*, 261–268, 1977.

[HPa] Harel, D. and M.S. Paterson, Undecidability of PDL with $L = \{ a^{2^i} \mid i \geq 0 \}$. RJ 4066, 10/83, IBM Research, San Jose. Submitted for publication.

[HP] Harel, D. and A. Pnueli, Two Dimensional Temporal Logic, Manuscript, 1982.

[HPS] Harel, D., A. Pnueli and J. Stavi, Propositional Dynamic Logic of Nonregular Programs, *J. Comput. Syst. Sci.* **26** (1983), 222–243.

[HV] Harel, D. and M. Vardi, PDL with Intersection, Manuscript, 1982.

[H] Hoare, C.A.R., An Axiomatic Basis for Computer Programming, *Comm. Assoc. Comput. Mach.* **12** (1969), 576–580, 583.

[KMW] Kahr, A.S., E.F. Moore and H. Wang, Entscheidungsproblem Reduced to the $\forall\exists\forall$ Case, *Proc. Nat. Acad. Sci.* USA, **48**, (1962), 365–377.

[K] Keisler, J., *Model Theory for Infinitary Logic*, North Holland, 1971.

[La] Ladner, R., The Computational Complexity of Provability in Systems of Modal Propositional Logic, *SIAM J. Comp.* **6** (1977), 467–480.

[L1] Lewis, H.R., Complexity of Solvable Cases of the Decision Problem for the Predicate Calculus, *19th IEEE Symp. Found. Comput. Sci.*, 35–47, 1978.

[L2] Lewis, H.R., *Unsolvable Classes of Quantificational Formulas*, Addison-Wesley, 1979.

[LP] Lewis, H.R. and C.H. Papadimitriou, *Elements of the Theory of Computation*, Prentice-Hall, 1981.

[MP] Manna, Z. and A. Pnueli, Verification of Concurrent Programs: Temporal Proof Principles, *Lect. Notes in Comput. Sci.*, Vol. **131**, Springer-Verlag, 200–252, 1981.

[Me] Mendelson, E., *Introduction to Mathematical Logic*, van Nostrand Reinhold, 1964.

[M] Meyer, A.R., Private Communication, 1977.

[MS] Meyer, A.R. and L.J. Stockmeyer, Word Problems Requiring Exponential Time, *5th ACM Symp. Theory of Comput.*, 1–9, 1973.

[MSM] Meyer, A.R., R.S. Streett and G. Mirkowska, The Deducibility Problem in Propositional Dynamic Logic, *Lect. Notes in Comput. Sci.*, Vol. **125**, Springer-Verlag, 12–22, 1981.

[Pn] Pnueli, A., The Temporal Logic of Programs, *18th IEEE Symp. Found. Comput. Sci.*, 46–57, 1977.

[P] Pratt, V.R., Semantical Considerations on Floyd-Hoare Logic, *17th IEEE Symp. Found. Comput. Sci.*, 109–121, 1976.

[RS] Reif, J.H. and A.P. Sistla, A Multiprocess Network Logic with Temporal and Spatial Modalities, TR-29-82, Harvard Univ., Computation Lab., 1982.

[Ro] Robinson, R.M., Undecidability and Nonperiodicity for Tilings of the Plane, *Inventiones Math.* **12**, (1971), 177–209.

[R] Rogers, H., *Theory of Recursive Functions and Effective Computability*, McGraw-Hill, 1967.

[Sa] Savitch, W.J., Relationships Between Nondeterministic and Deterministic Tape Complexities, *J. Comput. Syst. Sci.* **4** (1970), 177–192.

[Sh] Shoenfield, J.R., *Mathematical Logic*, Addison-Wesley, 1967.

[SC] Sistla, A.P. and E.M. Clarke, The Complexity of Propositional Linear Temporal Logics, *14th ACM Symp. Theory of Comput.*, 159–167, 1982.

[St] Stockmeyer, L.J., The Polynomial Time Hierarchy, *Theoret. Comput. Sci.*, **3** (1976), 1–22.

[S] Streett, R.S., Global Process Logic is Π_1^1-Complete, Manuscript, 1982.

[T] Turing, A.M., On Computable Numbers with an Application to the Entscheidungsproblem, *Proc. London Math. Soc. 2*, **42** (1936–7), 230–265, **43** (1937), 544–546.

[W1] Wang, H., Proving Theorems by Pattern Recognition II, *Bell Syst. Tech. J.* **40**, (1961), 1–41.

[W2] Wang, H., Dominoes and the AEA Case of the Decision Problem, In: *Mathematical Theory of Automata*, Polytechnic Press, 1963, 23–55.