LA-UR 95-4412

RECEIVED FEB 15 1996 OSTI

CONF-960599--1

TITLE: **PERIODICALLY SPECIFIED SATISFIABILITY PROBLEMS WITH APPLICATIONS: AN ALTERNATIVE TO DOMINO PROBLEMS**

AUTHOR(S): M. V. Marathe, H. B. Hunt III, D. J. Rosenkrantz, R. E. Stearns, V. Radhakrishnan

SUBMITTED TO: 11th Annual IEEE Conference on Computational Complexity
Philadelphia, PA
May 1996

Periodically Specified Satisfiability Problems with Applications: An Alternative to Domino Problems

Madhav V. Marathe[1] *Harry B. Hunt III*[2] *Daniel J. Rosenkrantz*[2]
Richard E. Stearns[2] *Venkatesh Radhakrishnan*[2,3]

November 13, 1995

Abstract

We characterize the complexities of several basic generalized CNF satisfiability problems SAT(S) [Sc78], when instances are specified using various kinds of 1- and 2-dimensional periodic specifications [Or82a, Wa93, HW94, HW95, CM91, CM93]. We outline how this characterization can be used to prove a number of new hardness results for the complexity classes DSPACE(n), NSPACE(n), DEXPTIME, NEXPTIME, EXPSPACE etc. The hardness results presented significantly extend the known hardness results for periodically specified problems [Or82a, Pa94, Wa93, HW94, HW95]. Several advantages are outlined of the use of periodically specified satisfiability problems over the use of domino problems in proving both hardness and easiness results.

As one corollary, we show that a number of basic NP-hard problems become EXPSPACE-hard when inputs are represented using 1-dimensional infinite periodic **wide** specifications. This answers a long standing open question posed by Orlin [Or82a].

[1]Current Address: Los Alamos National Laboratory P.O. Box 1663, MS K990 Los Alamos NM 87545. Email: madhav@c3.lanl.gov. The work is supported by the Department of Energy under Contract W-7405-ENG-36.

[2]Department of Computer Science, University at Albany - SUNY, Albany, NY 12222. Email addresses of authors: {hunt,djr,res}@cs.albany.edu. Supported by NSF Grants CCR 90-06396 and CCR94-06611.

[3]Current Address: Mailstop 47LA, Hewlett-Packard Company, 19447 Pruneridge Avenue, Cupertino, California 95014-9913. Email: rven@cup.hp.com

1 Introduction

Periodic specifications can be used to define large scale systems with highly regular structures. Using periodic specifications, large objects are described as repetitive connection of a basic module. Frequently, the modules are connected in a straight line but the basic modules can also be repeated in two or higher dimensional patterns. Periodic specifications have applications in such diverse areas as transportation planning [Or82a, HLW92, Ma94, HT95], parallel programming [HLW92, KMW67, HW94, HW95] and VLSI design [IS87, IS88]. They model periodic problems where the constraints or demands for any one period is the same as those for preceding or succeeding periods. Orlin has studied periodically specified problems over an infinite horizon; i.e the objects considered are infinite [Or82a]. Wanke [Wa93] and Hoppe and Tardos [HT95] have studied 1-dimensional wide specifications. Ford and Fulkerson [FF58] and Wanke [Wa93] have studied the finite horizon versions of periodic problems and Gale [Ga59] has studied problems for periodic specifications with fixed starting points. All the above mentioned studies are for 1-dimensional periodic specifications. Other researchers have studied 2-dimensional and more generally d-dimensional periodic specifications. (See [CM91, IS87, KO91, KS88, Wa93, HW94, HW95].) In [Or82a, Pa94], the problem 3SAT, for 1-dimensional infinite narrow periodically specified formulas (denoted here by 1-PN-3SAT), is defined and shown to be PSPACE-complete. Apart from this single result, the complexity of periodically specified generalized satisfiability problems have not been studied previously.

Domino (or Tiling) problems were introduced by Wang [Wa61] and Büchi [Bu62] and have been studied extensively in the literature. (see [vEB83, SB84, Ha85, Ha86, Fu84] and the references there in.) They have proven useful in obtaining hardness results, especially for decision problems for various logical theories. Usually a *domino system* is described as a finite set of *tiles* or *dominoes*, every tile being a unit square with a fixed orientation and colored edges; we have an unlimited supply of copies of every tile. A *domino problem* asks whether it is possible to tile a prescribed subset of the cartesian plane with elements of a given domino system, such that adjacent tiles have matching colors on their common edge and perhaps with certain constraints on the tiles that are allowed on certain specific places (e.g. the origin).

2 Summary of results

We define and study the complexity of generalized CNF satisfiability problems specified using various different kinds of periodic specifications. Our study generalizes Orlin's result [Or82a] that the problem 1-I(**Z**)PN-3SAT is PSPACE-complete and Schaefer's characterization [Sc78] of the complexity of generalized CNF satisfiability problems SAT(S), where S is a finite set of finite arity Boolean relations. Here, we consider the complexity of these problems SAT(S), when instances are specified using different kinds of periodic specifications. These kinds of specifications depend upon the answers to the following questions:
(1) Is the specified instance **1- or 2-dimensional** ?
(2) Is the specified instance **finite or infinite** ?
(3) Are specifications **narrow or wide** ?
(4) Are **explicit boundary conditions** allowed in the specifications ?
(5) Are bounds on finite dimensions specified in **unary (U)** or **binary (B)** ?
(6) Do the infinite dimensions range over **natural numbers (N)** or **integers (Z)** ?
A summary of our results, for the two problems 3SAT and 3SATWP appears in Table 1. Using the notation of Schaefer [Sc78], all of the hardness results for the problem 3SAT, also hold for each of the problems SAT(S) and $\mathsf{SAT}_c(\mathsf{S})$ shown to be NP-complete in [Sc78].

We can show that efficient reductions involving *local replacement* (possibly augmented with fixed size enforcers) [GJ79] of the problem 3SAT 1-3SAT, NAE-3SAT 3SATWP[4] etc, to a problem Π can be extended to obtain efficient reductions of the problems 3SAT 1-3SAT, NAE-3SAT 3SATWP, etc, to the problem Π, when instances are specified using the kinds kinds of periodic specifications considered here. These problems include most of the basic problems in [Ka72, MS81, GJ79] as well as several basic P-complete problems [JL77]. These results yield a number of new hardness results for the complexity classes DSPACE(n), NSPACE(n), DEXPTIME, NEXPTIME, EXPSPACE etc. depending on the kind of periodic specification used. To our knowledge, previously no DEXPTIME, NEXPTIME, EXPSPACE-hardness or undecidability results were known for periodically specified problems. We also note that since 2-dimensional finite periodic specifications can be viewed as simple types of S.C.R. specifications, our hardness for problems specified using 2-dimensional finite periodic narrow specfications, strengthen the hardness results in [PY86, BLT92][5] Similarly, our undecidability results for 2-dimensional infinite periodically specified satisfiability problems imply a number of undecidability results for problems specified using recursive graph specifications. Since 2-dimensional infinite periodic graph specifications are clearly a simple type of recursive graph specifications, these results strengthen several undecidability results in [Ha91, BG89, HH93].

As one corollary, we prove the EXPSPACE-hardness of a large class of combinatorial problems when specified by 1-dimensional wide periodic specifications. answering the following open question posed by Orlin [Or82a]: *"It is an interesting open question as to whether the non-narrow periodic graph problems are in the class* PSPACE.*"*

3 Advantages of a Theory Based on Periodically Specified Satisfiability Problems

We present several advantages of the use of periodically specified satisfiability problems over the use of domino problems in proving both hardness and easiness results.

1. Quoting Harel [Ha85]
   "Since all domino problems owe their complexity to the correspondence with Turing machine computations and since this correspondence applies to non-deterministic models as well, domino problems can apparently not distinguish between deterministic and non-deterministic classes." In contrast, the hardness results for periodically specified generalized CNF satisfiability problems include complete problems for the deterministic classes P, DSPACE(n), DEXPTIME, DEXPSPACE(n), etc.

   For example, our hardness results for 3SATWP, when instances are specified periodically with explicit boundary conditions, imply that exactly analogous hardness results hold for the Monotone circuit value problem, when instances are periodically specified with boundary conditions. The last result can be used to prove that a number of P-complete problems become DSPACE(n)-, DEXPTIME-, or DEXPSPACE-complete when periodically specified. One such problem is Linear programming feasibility.

2. We can show that the Horn formula satisfiability problem is *polynomial time solvable* or *undecidable* depending whether the problem is specified using periodic specifications with or without explicit boundary conditions. In contrast, for all the domino problems mentioned in

[4] Horn formula satisfiability problem is the restriction of the problem 3SAT, in which each clause has at most one *positive literal.* This is the same as the problem SATWN, studied by [Sc78]. The problem 3SATWP is similar to 3SATWN except that each clause has at most one negated literal.

[5] Although we strengthen several of the results in [PY86, BLT92], we do not have general metatheorems such as those given in [PY86, BLT92].

[Ha85, Ha86, vEB83, SB84], there is no difference (in terms of hardness) between the complexities of these domino problems with or without explicit boundary conditions.

3. It is natural to consider periodically specified formulas with clauses containing variables defined at times $t, t + c_1, t + c_2$, etc, where c_1 and c_2 are integers specified using binary numerals. Following Orlin [Or82a], we say that such periodic specifications are **wide**. Periodic specifications only containing clauses in which all the variables are defined at times $t, t + 1$ and $t - 1$ are called **narrow**. In contrast, domino problems are based on adjacency, and thus, are intrinsically narrow. Our hardness results for wide periodically specified satisfiability problems imply exactly analogous results for a number of problems specified using periodic wide specifications. Furthermore, these results show that there can be a significant difference between the complexities of the narrow and wide periodically specified versions of the same problem.

4. Efficient approximation preserving local replacement type reductions to/ from the problems MAX 3SAT, MAX NAE3SAT, MAX 2SAT can be extended to efficient approximation preserving reductions to/from MAX 3SAT, MAX NAE3SAT, MAX 2SAT, when instances are specified by various kinds of periodic specifications considered here. These reductions together with our easiness/hardness results imply analogous easiness/hardness results for a number of optimization problems for periodically specified graphs, etc. As one application we get the first collections of **natural** DEXPTIME-, NEXPTIME- and EXPSPACE-hard optimization problems with provably good polynomial time approximation algorithms approximation schemes [MHR95].

5. We can show that periodically specified versions of the problems 3SATWP and 3SAT are efficiently reducible to a number of problems Π when instances are specified succinctly as in [Ga82, GW83, PY86, BOW83, HLW92]. This yields unified proofs of the hardness of the problems Π when so specified [MHR95a]. For example, we can give direct efficient reductions of the PSPACE-hard problems 3SATWP or 3SAT specified using 1-dimensional finite periodic specifications with boundary conditions to each of the problems Π specified using the hierarchical specifications of Lengauer et al. [HLW92], shown to be PSPACE-complete in [LW92].

Our hardness results for periodically specified satisfiability problems are obtained by a direct reduction from the acceptance problem of time and space bounded Turing machines. The NEXPTIME-hardness of the problem 2-F(B,B)PN-3SAT (problem 3SAT specified using 2-dimensional finite periodic narrow specifications with both bounds in binary i.e. 2-F(B,B)PN-specifications) can also be proven by a reduction from a NEXPTIME-hard domino problem [Ha85, vEB83, SB84]. We give a direct proof since it it illustrates the underlying ideas used to prove all of the PSPACE-, DEXPTIME-, NEXPTIME- and EXPSPACE-hardness results in this paper. These hardness results include the hardness results for deterministic complexity classes DSPACE(n), DEXPTIME and the hardness results for problems specified using periodic wide specifications. As discussed above, these last results do not follow obviously from known hardness results for domino problems.

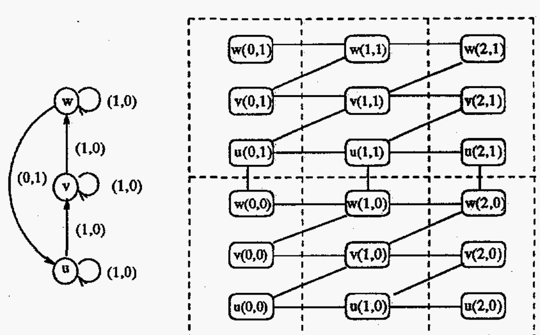
Due to lack of space, the remainder of this paper consists of preliminary definitions and selected proof sketches.

4 Preliminary Definitions

We first review, the concept of periodically specified instances. In what follows we discuss the concept of 2-dimensional periodically specified satisfiability problems. The notion of periodically specified graphs is given in [CM93, CM91, Wa93, HW94, HW95]. Figure 1 shows an example of periodic specification and the associated expanded graph.

Problems		Specification Type					Complete
Type	Name	Dimension(s) X,Y	Finite (F) or Infinite (I)	Boundary Conditions Present Yes (Y) or No (N)	Edges Narrow (N) or Wide (W)	F/I Bounds F,Unary (U) F, Binary (B) I, Int. (Z) I, Nat. (N)	
	1-F(B)PN(BC)-3SAT	1	F	Y	N	B	NSPACE(n)-C
	1-F(B)PN-3SAT	1	F	N	N	B	NSPACE(n)-C
	1-I(Z)PN-3SAT	1	I	N	N	Z	NSPACE(n)-C
	1-I(N)PN-3SAT	1	I	N	N	N	NSPACE(n)-C
	1-F(B)PW(BC)-3SAT	1	F	Y	W	B	NEXPTIME-C
	1-F(B)PW(BC)-3SAT	1	F	N	W	B	NEXPTIME-C
	1-I(ZPW-3SAT	1	I	-	W	Z	EXPSPACE-C
	2-F(B, U)PN-3SAT	2	F,F	-	N,N	B, U	NSPACE(n)-C
3SAT	2-F(B, B)PN-3SAT	2	F,F	-	N,N	B, B	NEXPTIME-C
	2-I(Z, B)PN-3SAT	2	I,F	-	N,N	B, Z	NEXPTIME-C
	2-I(Z, Z)PN-3SAT	2	I,I	-	N,N	Z,Z	undecidable
	2-I(N, N)PN-3SAT	2	I,I	-	N,N	N,N	undecidable
	1-F(B)PN-3SATWP	1	F	N	N	B	Poly
	1-I(Z)PN-3SATWP	2	I	N	N	Z	Poly
	2-I(Z, Z)PN-3SATWP	2	I	N	N	Z,Z	Poly
	2-I(N, N)PN-3SATWP	2	I	N	N	N,N	Poly
3SATWP	1-F(B)PN(BC)-3SATWP	1	F	Y	N	B	DSPACE(n)-C
	1-F(B)PW(BC)-3SATWP	1	F	Y	W	B	DEXPTIME-C
	2-F(B, U)PN(BC)-3SATWP	2	F,F	Y	N,N	B,U	DSPACE(n)-C
	2-F(B,B)PN(BC)-3SATWP	2	F,F	Y	N,N	B,B	DEXPTIME-C
	2-F(Z, B)PN(BC)-3SATWP	2	I,F	Y	N,N	Z,B	EXPSPACE-C
	2-F(N, N)PN(BC)-3SATWP	2	I,I	Y	N,N	N, N	undecidable

Table 1: Table summarizing the results for the problems 3SAT and 3SATWP when instances are specified using various kinds of periodic specifications. For example, the 8th row in the table specifies that the problem 3SAT when specified using 2-dimensional finite periodic narrow specifications, with the bounds on the X-axis specified in binary and the bounds on the Y-axis specified in unary is PSPACE-complete. **Z**, **N** stand for integers and natural numbers respectively.



(a) (b)

Figure 1: (a)The static graph with 2-dimensional integer vectors associated with each edge. (b) The graph $G^{2,1}$ specified by $\Gamma = (G, 10, 01)$.

For the rest of the paper, let **Z** and **N** denote the set of integers and natural numbers respectively.

Let $U = \{u_1, \ldots, u_n\}$ be a finite set of variables (referred to as static variables). $U^{M,N} = \{u_k(i,j) : 1 \le k \le n,\ i \in \{0,1,2,\cdots,M\},\ j \in \{0,1,2,\ldots,N\}\}$. (In our proofs, variable $u_k(i,j)$ denotes the variable u_k at grid point (i,j).) A literal of U is an element of $\{u_1,\ldots,u_n,\overline{u_1},\ldots,\overline{u_n}\}$. If w is a literal of U, then $w(i,j)$, $0 \le i \le M$ and $0 \le j \le N$ is a literal of $U^{M,N}$. Let $C(i,j,i+1,j+1)$ be a parameterized conjunction of 3 literal clauses such that each clause in $C(i,j,i+1,j+1)$ consists of variables $u_k(i,j), u_k(i+1,j), u_k(i,j+1), u_k(i+1,j+1)$ with the constraint that at least one variable is of the form $u_k(i,j)$. We refer to the clauses $C(i,j,i+1,j+1)$ as *static narrow clauses.* ($C(i,j)$ is called narrow because for all $(w_1(i_1,j_1) \vee w_2(i_2,j_2) \vee w_3(i_3,j_3)) \in C(i,j,i+1,j+1)$, $|i_s - i_r|, |j_s - j_r| \le 1$ for $1 \le r \le s \le 3$.) The conjuction of static narrow clauses is referred to as static narrow formula. Let $\Gamma = (U, C(i,j,i+1,j+1), M, N)$. Let $\mathcal{C} = \bigwedge_{i=0,j=0}^{i=M,j=N} C(i,j,i+1,j+1)$. Then $\mathcal{C}$ is the 3CNF formula specified by Γ. Given $U^{M,N}$ and $\mathcal{C}$, let $C^{M,N}$ be a subset of $\mathcal{C}$ with the following property: for each clause $(w_1(i_1,j_1) \vee w_2(i_2,j_2) \vee w_3(i_3,j_3)) \in C^{M,N}$, $w_1(i_1,j_1), w_2(i_2,j_2), w_3(i_3,j_3) \in U^{M,N}$.

Definition 4.1 A 2-dimensional finite periodic narrow specification (2-F(B,B)PN-specification) of a 3CNF formula $F^{M,N}(U^{M,N}, C^{M,N})$ is a four tuple $\Gamma = (U, C(i,j,i+1,j+1), M, N)$, where, U is a a finite set of variables, $C(i,j)$ is a collection of static narrow 3 literal clauses, and M, N are non-negative integers specified in binary. The size of the specification denoted by $size(\Gamma) = |U| + |C(i,j,i+1,j+1)| + bits(M) + bits(N)$, where $bits(M)$ and $bits(N)$ denote the number of bits used to represent M and N respectively.

The problem 2-F(B,B)PN-3SAT (problem 3SAT specified using 2-dimensional finite periodic narrow specifications with both bounds in binary i.e. 2-F(B,B)PN-specifications) is the problem of determining if a 3CNF formula $F^{M,N}(U^{M,N}, C^{M,N})$ specified by $\Gamma = (U, C(i,j,i+1,j+1), M, N)$ is satisfiable.

2-I(**Z**, B)PN-3SAT is the problem of, given a 2-dimensional periodic specification $\Gamma = (F(U, C(i,j,i+1,j+1), m)$, where m denotes the width (in terms of Y-axis) is specified in binary, determining whether the CNF formula, $\bigwedge_{i=-\infty,j=0}^{i=\infty,j=m} C(i,j,i+1,j+1)$ satisfiable.

4.1 Note on Naming Convention

Since we have a large number of parameters, it is necessary to state the notation used throughout this abstract for naming problems. We use F and I to denote *finite* or *infinite* graphs respectively. Observe that while this is the property of the expanded object, we choose to use this as a way to classify the specification itself. The symbols U, B in the brackets following F specify, whether the finite bounds are specified in unary or binary notation. The symbols **N**, **Z** following I specify whether the graph is inifinite in one direction or both the directions. We have already explained the concept of narrow and wide specifications. We use N and W to denote **narrow** and **wide** specifications respectively. *Dimensions of the expanded Graph:* $\{1,2,\ldots d\}$ denote the dimensions in which the static graph is translated. Some instances of problems arising in practice have a periodic specification of the graph or a formula along with explicit initial and final conditions. We call such periodic specifications as periodic specifications with boundary conditions (BC).

Example 1: Let the set of static variables $U = \{x, y, z\}$. The static clauses C is specified by $C(i,j,i+1,j+1) = [x(i,j) + y(i,j) + z(i,j)] \wedge [x(i+1,j) + y(i,j) + z(i+1,j)] \wedge [x(i,j+1) + z(i,j)]$. The set of clauses $C^{1,1}$ is given by

$[x(0,0)+y(0,0)+z(0,0)] \wedge [x(0,1)+y(0,1)+z(0,1)] \wedge [x(1,0)+y(1,0)+z(1,0)] \wedge [x(1,1)+y(1,1)+z(1,1)] \wedge [x(1,0)+y(0,0)+z(1,0)] \wedge [x(1,1)+y(0,1)+z(1,1)] \wedge [x(0,1)+z(0,0)] \wedge [x(1,1)+z(1,0)]$

5 Selected Proofs

5.1 Basic Technique for Proving Hardness Results

The main idea involves the construction of a static formulas, force the satisfiability of the expanded formulas to correspond to the existence of legal computations of Turing machines. Intuitively, we have one column for each step of a computation and one row for each tape cell of the Turing machine. Proving hardness results for satisfiability problems without explicit boundary conditions is subtle, since there is no obvious way to force the the Turing machine to start correctly.

5.2 NEXPTIME-completeness of 2-F(B,B)PN-3SAT

Theorem 5.1 (1) 2-F(B,B)PN-3SAT is NEXPTIME-complete.

Proof of Theorem 5.1: Membership in NEXPTIME follows easily by observing that the size of the expanded formula is $2^{c(|U+C(i,j,i+1,j+1)|+bits(M)+bits(N))}$, where $\Gamma = (U, C(i,j), M, N)$, is the specification of $F^{M,N}$. Hence a NEXPTIME bounded TM can guess an assignment to the variables and then verify in DEXPTIME that the assignment satisfies all the clauses.

Next, we discuss the reduction which shows the NEXPTIME-hardness of the problem. It is worth pointing out the basic technique used behind the reductions. Since the static formula associated with 2-FPN-3SAT instance is the same for each time period, it is not possible to write a 3CNF formula which says that the machine has the correct starting ID. This makes the task of constructing the 3SAT instance more difficult. In order to overcome this difficulty, our reduction consists of two phases. In the first phase, we start with a given Turing machine ϕ with input $x = (x_1, \ldots, x_n)$ and construct a new Turing machine ϕ_x which simulates ϕ on x and has the following additional properties that

1. If Turing machine ϕ does not accept x, then every possible computation of ϕ_x halts within $2^{c_0 n}$ moves, else

2. If Turing machine ϕ accepts x, then ϕ_x has a cycling computation, where the length of an ID never exceeds $2^{d_0 n}$, for some given d_0.

The second phase consists of constructing an instance $(U_x(t,y), G_x(t,y,t+1,y+1), M, N))$ of 2-FPN-3SAT by a polynomial time reduction from ϕ_x. Now we know that each ID of the Turing machine ϕ_x is of length $2^{d_0 n}+1$. From Property 2 above, we need to consider only $2^{d_0 n}$ different ID's for our reduction. In order to understand the construction imagine each ID of the Turing machine ϕ_x being placed vertically in the plane. Two consecutive ID's of ϕ_x are placed vertically next to each other. For the sake of exposition we will refer to the X-axis as the *time line.* In the following discussion, each grid point is referred to as (t, y). We now define the set of variables $U_x(t,y)$ and their intended meaning. $U_x(t,y)$ consists of the following three different types of variables. (i) $TAPE \subset U_x(t,y)$, such that $TAPE(t,y)$ encodes the y^{th} symbol in the t^{th} ID. $TAPE(t,y)$ takes values from the set $\{\#\} \cup \Gamma \cup (Q \times \Gamma)$, where Γ denotes the tape symbols and Q denotes the set of states of ϕ_x. The number of variables needed to encode $TAPE(t,y)$ depends only on the machine ϕ_x. (ii) In order to simulate the behavior of ϕ_x properly we need to have two set of counter variables; c_y and c_t. The counter c_y keeps track of the particular tape cell in a given ID. Let $q = d_0 n$. The counter can be simulated by means of $(d_0 n + 1)$ Boolean variables $tc_q, tc_{q-1}, \ldots tc_0$. tc_0 represents the least significant bit and tc_q represents the most significant bit. The counter c_t keeps track of the number of ID's. The counter c_t can be simulated by means of $(d_0 n + 1)$ Boolean variables. We use Boolean variables $yc_0, yc_1, yc_2, \ldots, yc_q$ to simulate the counter c_y. (iii) Auxiliary variables for making the resulting static formula narrow and in the 3CNF form.

The initial ID is of the form $\#(q_0,x_1)\ldots x_n B^{2^{d_0 n}-n}$, where B denotes a blank. The static formula CNF formula $G_x(t,y)$ is given by $G_x(t,y) = f_1(t,y) \wedge f_2(t,y) \wedge f_3(t,y)$. We now describe each of the subformulas $f_i, 1 \le i \le 3$ separately. Each f_i is described in terms of variables at coordinates $y, y+1, t, t+1$.

Counter Updating: Formula f_1

$f_1 \equiv f_1^1 \wedge f_1^2 \wedge f_1^3 \wedge f_1^4$, where each f_1^i, $1 \le i \le 3$ is given as follows:
$f_1^1 \equiv [c_t(t+1,y) = (c_t(t,y)+1) \pmod{2^{d_0 n}+1},]$ $\quad f_1^2 \equiv [0 \le c_y(t,y) < 2^{d_0 n} \Rightarrow c_t(t,y+1) = c_t(t,y)]$
$f_1^3 \equiv [c_y(t,y+1) = (c_y(t,y)+1) \pmod{2^{d_0 n}+1}]$ $\quad f_1^4 \equiv [0 \le c_t(t,y) < 2^{d_0 n} \Rightarrow c_y(t+1,y) = c_y(t,y)]$

f_1^1 says that the value of the counter c_t at grid point $(t+1,y)$ is 1 more than the value of the counter at the grid point (t,y). Moreover, the counter is reset after every $2^{d_0 n}+1$ time units. f_1^2 says that the counter value for a given value of t is the same for all y. Conjuncts f_1^3 and f_1^4 describe the desired properties of the counter c_y in a manner similar to f_1^1 and f_1^2.

Implicit Initialization: Formula f_2

$[(c_y(t,y) = 0) \wedge (c_t(t,y) = 0) \Rightarrow TAPE(t,y) = \#] \wedge [(c_y(t,y) = 1) \wedge (c_t(t,y) = 0) \Rightarrow TAPE(t,y) = (q_0 x_1)] \wedge$

$\vdots$

$[(c_y(t,y) = n) \wedge (c_t(t,y) = 0) \Rightarrow TAPE(t,y) = x_n] \wedge [(n+1 \le c_y(t,y) \le 2^n) \wedge (c_t(t,y) = 0) \Rightarrow TAPE(t,y) = B]$

f_2 can be thought of as a way to implicitly initialize the clauses to reflect that the machine starts out right whenever the counters are reset to 0. The initialization condition say that if both the counter values are 0, then we have # as the tape symbol and so on.

Consistency Checking: Formula f_3

$(0 \le c_y(t,y) \le 2^{d_0 n}) \wedge (2^n + 1 \le c_t(t,y) \le 2^{d_0 n}) \Rightarrow$
$Consistent(TAPE(t,y), TAPE(t,y+1), TAPE(t,y+2), TAPE(t+1,y), TAPE(t+1,y+1), TAPE(t+1,y+2))$

f_3 ensures the consistency of the tape symbols, i.e. that the contents of the tape cells i, $i+1$ and $i+2$ in ID_t determine the contents of the tape cells i, $i+1$ and $i+2$ in ID_{t+1}. The Consistency function of course depends on the state transition relation.

Although, the above formula contains clauses which are not narrow, it is easy to transform them to a narrow set of clauses by adding temporary variables. We omit the details in this abstract. Now, it is easy to see that these equations can be transformed into an equivalent narrow 3CNF formula $G_x(t,y)$ whose size is polynomial is in n, (recall that $n = |x|$.) The expanded finite periodic 3SAT instance is $\bigwedge_{y=0,t=0}^{y=N,t=M} G_x(t,y,t+1,y+1)$, where $M = 2^{2d_0 n}$ and $N = 2^{2d_0 n}$.

We now prove the correctness of our reduction. If the Turing machine ϕ accepts x then we know that ϕ_x has a cycling computation. Hence by setting the counters $c_t(0,0) = c_y(0,0) = 0$ we get that the first column of the grid contains the right initial ID. From then on, the consistency conditions ensures that the formula $\bigwedge_{y=0,t=0}^{y=N,t=M} G_x(t,y,t+1,y+1)$ is satisfied. Conversely, assume that the formula is satisfiable. Since M and N are suitably large integers, it is guaranteed that the following two conditions hold:

1. Since N is large enough, the simulation must be carried out for enough steps so that the Turing machine ϕ_x goes through the sequence $c_t = 0, c_t = 1, c_t = 2, \cdots c_t = 2^{d_0 n}$. This implies that the formulas $f_2(t,y)$ and $f_3(t,y)$ would be true from the time when the value of $c_t = 0$.

2. Similarly, since M is large enough, the grid is sufficiently long in the Y-direction so that the counter value c_y goes through a sequence of values $c_y = 0, c_y = 1, c_y = 2, \cdots c_y = 2^{don}$. This implies that the first part of the implication in f_2 is true and from then on, it is ensured that the TM ϕ_x goes through the simulation correctly.

The above two conditions imply that if the formula $\bigwedge_{y=0,t=0}^{y=N,t=M} G_x(t, y, t+1, y+1)$ is satisfied then the Turing machine ϕ accepts x. ■

5.3 Proving EXPSPACE-hardness of 2-I(N,B)PN-3SAT

Although there are technical difficulties, the basic idea behind the proof is similar to the idea used to prove NEXPTIME-hardness of 2-F(B,B)PN-3SAT. Therefore, we only point out essential differences. Recall that we used two counters to keep a track of the length of each ID and also to keep track of the number of ID's. Since in the proofs of NEXPTIME-hardness, we need only consider singly exponential many ID's we were able to use a counter which had only polynomial number of bits. This in particular implied that the variables constituting the counter can occur together **explicitly** in the static formula. In this case, we want to simulate an 2^{cn} space-bounded Turing machine and this means that we need to keep track of roughly $2^{2^{cn}}$ ID's. To do this we need a counter with roughly 2^{cn} bits. Thus all the variables constituting the counter cannot occur together **explicitly** in the static formula.

It is easy to see that the above ideas can be extended to prove hardness results for wide specifications. We only give the basic idea behind the reduction. For this, it is useful to imagine each ID being rotated horizontally on the X-axis. Now observe that the narrow clauses used to describe the relationship between variables of consecutive ID's in case of 2-dimensional specifications can be replaced by wide clauses in the 1-dimensional specifications. Summarizing the discussion, we get the following theorem.

Theorem 5.2 (1) The problems 2-I(**Z**, B)PN-3SAT, 1-I(**Z**)PW-3SAT[6] are EXPSPACE-hard.
(2) The following problems are EXPSPACE-complete when instances are specified by 2-I(**Z**, B)PN or 1-I(**Z**)PW specifications: independent set, 3 coloring, dominating set, vertex cover, partition into triangles.

5.4 Undecidability of 2-I(N,N)PN(BC)-3SATWN

Instance: A deterministic Turing machine M and a string x over Σ^*.
Question; Does M diverge on x ?
It is well known that T1 is undecidable. We use this problem to show that 2-I(**N, N**)(BC)-3SATWN is undecidable.

Theorem 5.3 2-I(**N, N**)PN(BC)-3SATWN is undecidable.

Proof Sketch: The reduction is from the problem T1. Given an instance of the problem T1, we construct an instance of the problem 2-I(N,N)PN(BC)-3SATWP as follows: Let $ID(t)$ denote the instantaneous description of a a deterministic Turing machine M at time time t. Given M and input $x = (a_1, \ldots a_n)$ such that $|x| = n$ we create an instance $F_x(t, t+1, y, y+1)$ of 1-FPN(BC)-3SATWN, such that $\bigwedge_{t=0,y=0}^{t=\infty,y=\infty} F_x(t, t+1, y, y+1)$ is satisfiable if and only if M diverges on x. The formulas encoding $(ID(t) \vdash ID(t+1))$, and $start(ID(t))$ can be represented by a weakly negative formula as in the P-hardness of UNIT given in [JL77]. Let $a \in \{\#\} \cup T \cup (S \times T)$, where T denotes the tape symbols and S denotes the set of states. Let $T_1 = T - \{B\}$ (B denotes a blank). Let $P^a(t, y)$

[6]The problem 3SAT specified using 1-dimensional two way infinite periodic wide specifications.

be a Boolean variable which means that the contents of y^{th} tape cell at time t is a. The formula $g_x(0,y,y+1)$ is now represented as $g_x(0,y,y+1) \equiv B_1 \wedge B_2 \wedge B_3$ where each of the $B_i's$ are defined as follows:

$$B_1 \equiv \left(P^{(q_0,a_1)}(0,0) \wedge P^{a_2}(0,1), \ldots \wedge P^{a_n}(0,n-1)\right)$$

$$B_2 \equiv \left(\bigwedge_y \bigwedge_{a \neq b} (\overline{P^a(0,y)} \vee \overline{P^b(0,y)})\right) \quad B_3 \equiv \left(\bigwedge_{y=n+1}^{y=\infty} P^B(0,y) \bigwedge_{y=n+1,\ a \in T_1}^{y=\infty} \overline{P^a(0,y)}\right)$$

B_1 encodes the condition that the first ID corresponds to the input. B_2 encodes the fact that the first n tape cells do not contain 2 distinct symbols. B_3 encodes the condition that the remaining part of the tape is initialized with all blanks. Hence, the above formula represents the condition that the first ID is correct. Also observe that since the number of tape symbols are constant, the size of $g_x(0,y,y+1)$ is $O(n \log n)$.

Next, we represent $(ID(t) \vdash_M ID(t+1))$ as follows: Let $f : (T \cup (S \times T))^3 \rightarrow T \cup (S \times T)$ be the finite function such that if positions $i-1$, i and $i+1$ of the $ID(t)$ contain a, b and c respectively, then position i of the $ID(t+1)$ must contain $f(a,b,c)$. The determinism of M ensures that f is single valued. We express the requirement that $ID(t+1)$ is appropriately determined by $ID(t)$ as follows:

$$h_x(t,t+1,y,y+1) \equiv \bigwedge_{a,b,c \in T} \left(\left(P^a(t,y-1) \wedge P^b(t,y) \wedge P^c(t,y+1)\right) \Rightarrow P^{f(a,b,c)}(t+1,y)\right)$$

which can be written as an equivalent weakly negative formula by using auxiliary variables $T^{f(a,b,c)}(t,y)$ as follows:

$$\bigwedge_{a,b,c \in T} \left(\left(\overline{P^a(t,y)} \vee \overline{P^b(t,y)} \vee T^{f(a,b,c)}(t,y)\right) \wedge \left(\overline{T^{f(a,b,c)}(t,y)} \vee \overline{P^c(t,y+1)} \vee P^{f(a,b,c)}(t+1,y)\right)\right)$$

$F_x(t,t+1,y,y+1) = g_x(0,y,y+1) \cup h_x(t,t+1,y,y+1)$. The corresponding expanded formula is given as

$$F_x^{\mathbf{N},\mathbf{N}} = g_x(0,y,y+1) \wedge \bigwedge_{t=0,y=0}^{t,y \in \mathbf{N}} h_x(t,t+1,y,y+1).$$

Again observe that since the number of tape symbols are constant the size of the formula $h_x((t,t+1,y,y+1)$ is $(n \log n)$. It can now easily be verified that the expanded formula is satisfiable if and only if the Turing machine M starting on x has a divergent computation. This completes the proof of the theorem. ■

5.5 Polynomial time solvability of 2-F(Z,Z)PN-3SATWP

Next, we consider the problems 2-F(B,B)PN-3SATWP, 2-I(N,N)PN-3SATWP and 2-I(Z,Z)PN-3SATWP. Extending our results for these problems to similar problems involving Horn formula satisfiability is straightforward and is omitted here. In contrast to the undecidability of solving 2-I(N,N)PN(BC)-3SATWP, we show that each of the above three problems has a polynomial time algorithm. This points out a major difference between these variants of periodic specifications.

We first consider the problem 2-I(Z,Z)PN-3SATWP. Recall that a relation R is weakly positive if R is equivalent to some CNF formula having at most one negated variable in each conjunct. The algorithm for solving the problem 2-I(Z,Z)PN-3SATWP is relatively easy, and is based on the following two observations. The first observation is that if there is a clause with only one literal, all copies of the corresponding variable must have the same value. For instance, if there is a clause consisting of the single literal $\overline{x_i(t+1,y)}$, then all copies of variable x_i have to be set to false. The second observation is that after simplifying the set of clauses as much as possible on the basis of the first observation, every remaining clause has either no literals or more than one literal. Weak positivity implies that each clause with more than one literal contains at least one positive literal, so setting all remaining variables to true will satisfy all such clauses. Since each simplification of the set of clauses based on the first observation assigns a value to a variable in the static formula which has not been previously assigned a value, the algorithm will terminate in polynomial time. Note that if the expanded formula for the given instance of 2-I(**Z**,**Z**)PN-3SATWP is satisfiable, there exists a satisfying assignment that assigns the same value to all the copies of a given variable in the static formula.

Next consider the problems 2-I(**N**,**N**)PN-3SATWP and 2-F(B,B)PN-3SATWP. Any algorithm for solving these problems must deal with subtle issues created by the presence of a "boundary" in the expanded formula. A clause of the form $x_i(t,y)$ implies that x_i is set to true for all time periods. However, a clause of the form $x_i(t+1,y)$ does imply anything about the value of the variable $x_i(0,y)$ in a satisfying assignment of the formula. Similar arguments hold for clauses of the form $x_i(t,y+1)$ and $x_i(t+1,y+1)$ (the second clause might arise after the elimination of other variables.) The following simple example, shows that even for 1-dimensional specifications, there are cases where all satisfying assignments to the expanded formula assign different values to the copies of a particular variable.

Example 2: Let $F = (U, C(t,t+1), 1)$ be an instance of 1-F(B)PN-3SATWP where the set of static clauses are given by $(x_1(t) + x_2(t+1)) \wedge (\overline{x_2(t)}) \wedge (x_2(t) + \overline{x_1(t+1)})$. The set of variables are $U = \{x_1, x_2\}$. The expanded formula is $(x_1(0) + x_2(1)) \wedge (\overline{x_2(0)}) \wedge (x_2(0) + \overline{x_1(1)}) \wedge (\overline{x_2(1)})$.

By inspection it is clear that any assignment to the variables of the expanded formula such that $v[x_1(0)] = v[x_1(1)]$ and $v[x_2(0)] = v[x_2(1)]$ cannot satisfy the formula. But the assignment $v[x_1(0)] = 1$, $v[x_1(1)] = 0$, $v[x_2(0)] = v[x_2(1)] = 0$ satisfies the expanded formula. A similar example can be constructed for the 2-dimensional case. ∎

Example 2 suggests that a polynomial time algorithm for solving 1-I(**N**,**N**PN-3SATWP should distinguish between the copy of each variable at $t = 0, y = 0$ and the copies of the same variable at time $t, y \geq 0$. Our polynomial time algorithm for 2-I(**N**,**N**)PN-3SATWP considers four groups of variables, corresponding to $(t = 0, y = 0)$, $(t = 0, y > 0)$, $(t > 0, y = 0)$, and $(t > 0, y > 0)$.

Theorem 5.4 The problems 2-F(B,B)PN-3SATWP, 2-I(**N**,**N**)PN-3SATWP and 2-I(**Z**,**Z**)PN-3SATWP are in P.

6 Conclusions

Our results suggest the use of periodically specified satisfiability problems to prove hardness results for various kinds of temporal logics. We suspect that a number of new insights will result from this. For instance, we believe that the PSPACE-hardness results of Emerson and Sistla [SC85] on the complexity of linear propositional temporal logics, the EXPSPACE-hardness and the undecidability results of Alur and Henzinger [AH94], for temporal logics with freeze quantifiers, etc, can be intuitively explained by the results obtained here. The PSPACE-hardness is analogous to our PSPACE-hardness result for 1-dimensional periodically specified satisfiability problem. The EXPSPACE-hardness result of [AH94] intuitively comes from the fact that they allow wide edges in their specifications by means of freeze quantifiers and constants written in binary notation. This result is analogous to our EXPSPACE-hardness result for 1-dimensional wide periodically specified satisfiability problem.

Acknowledgements

We thank Professors A. Condon, T. Lengauer and E. Wanke for valuable discussions on periodic specifications. We also thank Professors van Emde Boas, M. Fürer, Prof. Orlin and Dr. E. Cohen for making available copies of their papers. Finally, we thank S. Shukla, R. Sundaram, and Professor S. S. Ravi for discussions, comments and their help throughout the course of writing this paper.

References

[AH94] R. Alur and T. Henzinger, "A Really Temporal Logic," *Journal of the ACM (J. ACM)* Vol. 41, No. 1, January 1994, pp. 181-204.

[BOW83] J.L. Bentley, T. Ottmann and P. Widmayer, "The Complexity of Manipulating Hierarchically Defined set of Rectangles," *Advances in Computing Research, ed. F.P. Preparata* Vol. 1, (1983), pp. 127-158.

[BG89] R. Beigel, W. I. Gasarch, " On the Complexity of Finding the Chromatic Number of Recursive Graphs," Parts I and II, *Annals of Pure and Applied Logic*, 45, 1989, pp. 1-38 and 227-247.

[CM93] E. Cohen and N. Megiddo, "Strongly Polynomial-time and NC Algorithms for Detecting Cycles in Dynamic Graphs," *Proc. 21st ACM Annual Symposium on Theory of Computing (STOC)*, 1989, pp. 523-534. Journal version appears in *Journal of the ACM (J. ACM)* Vol. 40, No. 4, September 1993, pp. 791-830.

[CM91] E. Cohen and N. Megiddo, "Recognizing Properties of Periodic graphs", *Applied Geometry and Discrete Mathematics, The Victor Klee Festschrift* Vol. 4, P. Gritzmann and B. Strumfels, eds., ACM, New York, 1991, pp. 135-146.

[FF58] L.R. Ford and D.R. Fulkerson, "Constructing Maximal Dynamic Flows from Static Flows," *Operations Research,* No. 6, 1958, pp. 419-433.

[Bu62] J.R. Büchi, "Turing Machines and the Entscheidungsproblem," *Math. Ann.* 148, 1962, pp. 201-213.

[Fu84] M. Fürer, "The Computational Complexity of the Unconstrained Limited Domino Problem (with Applications for Logical Decision problems)," *Proc. Decision Problems and Complexity* LNCS Vol. 171, Springer Verlag, 1984, pp. 312-319.

[Ga59] D. Gale, "Transient Flows in Networks," *Michigan Mathematical Journal*, No. 6, 1959 , pp. 59-63.

[Ga82] H. Galperin "Succinct Representation of Graphs," Ph.D. Thesis, Princeton University, 1982.

[GW83] H. Galperin and A. Wigderson, "Succinct Representation of Graphs," *Information and Control* , Vol. 56, 1983, pp. 183-198.

[GJ79] M.R. Garey and D.S. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness*, Freeman, San Francisco CA, 1979.

[Ha86] D. Harel, "Effective Transformations on Infinite Trees, with Applications to High Undecidability, Dominoes, and Fairness," *Journal of the ACM (J. ACM)* , Vol. 33, No. 1, January, 1986, pp. 224-248.

[Ha85] D. Harel, "Recurring Dominos: Making Highly Undecidable Highly Understandable," *Annals of Discrete Math.* , Vol. 24, 1985, pp. 51-72. A preliminary version of the paper appeared in *Proc. Conference on Foundations of Computation Theory,* LNCS Vol. 158, pp. 177-194.

[Ha91] D. Harel, "Hamiltonian Paths in Infinite Graphs," *Israel J. Math.* 76: 3, 1991, pp. 317-336. A preliminary version of the paper appears in *Proc. 23rd Annual ACM Symposium on Theory of Computing, (STOC)*, 1991, pp. 220-229.

[HH93] T. Hirst, D. Harel, "Taking it to the Limit: On Infinite Variants of NP-Complete Problems," *Proc. 8th IEEE Annual Conference on Structure in Complexity Theory*, June, 1993, pp. 292-304.

[HLW92] F. Höfting, T. Lengauer and E. Wanke, "Processing of Hierarchically Defined Graphs and Graph Families," in *Data Structures and Efficient Algorithms* (Final Report on the DFG Special Joint Initiative), Springer-Verlag, LNCS 594, 1992, pp. 44-69.

[HW95] F. Höfting and E. Wanke, "Minimum Cost Paths in Periodic Graphs," *SIAM J. on Computing*, Vol. 24, No. 5, Oct. 1995, pp. 1051-1067

[HW94] F. Höfting and E. Wanke, "Polynomial Time Analysis of Toroidal Periodic Graphs," *Proc. of International Colloquium on Automata, Programming and Languages*, LNCS, Oct. 1994, pp.

[HT95] B. Hoppe, E. Tardos, "The quickest Transhipment Problem," *Proc. 6th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1995, pp. 512-521.

[Hu73a] H.B. Hunt III, "On The Time and Tape Complexity of Languages," Ph.D. thesis, Department of Computer Science, Cornell University, August, 1973.

[IS87] K. Iwano and K. Steiglitz, "Testing for Cycles in Infinite Graphs with Periodic Structure," *Proc. 19th Annual ACM Symposium on Theory of Computing, (STOC)*, 1987, pp. 46-53.

[IS88] K. Iwano and K. Steiglitz, "Planarity Testing of Doubly Connected Periodic Infinite Grpahs," *Networks*, No. 18, 1988, pp. 205-222.

[JL77] N.D. Jones and W.T. Laaser, "Complete Problems for Deterministic Polynomial Time," *Theoretical Computer Science*, No. 3, 1977, pp. 105-117.

[KMW67] R.M. Karp, R.E. Miller and S. Winograd, "The Organization of Computations for Uniform Recurrence Equations," *Journal of the ACM (J. ACM)*, Vol. 14, No. 3, 1967, pp. 563-590.

[Ka72] R.M. Karp, "Reducibility Among Combinatorial Problems," in R.E. Miller and J.W. Thatcher (eds) *Complexity of Computer Computations*, Plenum Press, N.Y. 1972, pp. 85-103.

[KO91] M. Kodialam and J.B. Orlin, "Recognizing Strong Connectivity in Periodic graphs and its relation to integer programming," *Proc. 2nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1991, pp. 131-135.

[KS88] K. R. Kosaraju and G.F. Sullivan, "Detecting Cycles in Dynamic Graphs in Polynomial Time," *Proc. 27th IEEE Symposium on Foundations of Computer Science (FOCS)*, 1988, pp. 398-406.

[LW92] T. Lengauer and K.W. Wagner, "The Correlation Between the Complexities of Non-Hierarchical and Hierarchical Versions of Graph Problems," *Journal of Computer and System Sciences (JCSS)*, Vol. 44, 1992, pp. 63-93.

[BLT92] J.L. Balcazar, A. Lozano, and J. Toran " The Complexity of Algorithmic Problems for Succinct Instances," in *Computer Science* Ed. R. Baeza-Yates, 1992, pp. 351-377. An earlier version of this paper appeared in *Proc. 15th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'89)* , Springer Verlag, LNCS, Vol. 411, 1989, pp. 277-286.

[MH+94] M.V. Marathe, H.B. Hunt III, R.E. Stearns and V. Radhakrishnan, "Approximation Schemes for PSPACE-Complete Problems for Succinct Specifications," *Proc. 26th ACM Annual Symposium on Theory of Computing (STOC)*, 1994, pp. 468-477.

[Ma94] M.V. Marathe *Complexity and Approximability of NP- and PSPACE-hard Optimization Problems,* Ph.D. thesis, Department of Computer Science, University at Albany, Albany, NY August, 1994.

[MHR95] M.V. Marathe, H.B. Hunt III, R.E. Stearns and D. J. Rosenkrantz, "Approximation Algorithms for NEXPTIME-Complete Periodically Specified Problems," submitted, 1995.

[MHR95a] M.V. Marathe, H.B. Hunt III and R.E. Stearns "A Uniform Approach to prove hardness for problems specified by G.C.R, S.C.R and BOW Specifications," manuscript, Nevmeber, 1995.

[MS81] B.Monien and I.H.Sudborough, "Bounding the Bandwidth of NP-Complete Problems," *Proc. 13th ACM Annual Symposium on Theory of Computing (STOC)*, 1981, pp. 279-292.

[Or82a] J.B. Orlin, "The Complexity of Dynamic/Periodic Languages and Optimization Problems," Sloan W.P. No. 1679-86 July 1985, Working paper, Alfred P. Sloan School of Management, MIT, Cambridge, MA 02139. A Preliminary version of the paper appears in *Proc. 13th ACM Annual Symposium on Theory of Computing (STOC)*, 1978, pp. 218-227.

[Or84b] J.B. Orlin, "Some Problems on Dynamic/Periodic Graphs," *Progress in Combinatorial Optimization*, Academic Press, May 1984, pp. 273-293.

[PY86] C. Papadimitriou and M. Yannakakis, "A note on Succinct Representation of Graphs," *Information and Computation* No. 71, 1986, pp. 181-185.

[Pa94] C. Papadimitriou, *Computational Complexity* Addison-Wesley, Reading, Massachusetts, 1994.

[Sc78] T. Schaefer, "The Complexity of Satisfiability Problems," *Proc. 10th ACM Symposium on Theory of Computing (STOC)*, 1978, pp. 216-226.

[SC85] A. P. Sistla and E. Clark, "Complexity of Linear Temporal Logics," *Journal of the ACM (J. ACM)* Vol. 32, No. 3, July 1985, pp. 733-749.

[SB84] M. Savelsberg, van Emde Boas, "Bounded Tiling, An Alternative to SATISFIABILITY," *Proc. 2nd Frege Conference*, G. Wechsung, Ed. Akademie Verlag, Berlin, DDR. 20, 1984, pp. 354-363.

[vEB83] van Emde Boas, "Dominoes Forever," *Proc. 1st GTI Workshop*, Paderborn, 1983, pp. 76-95.

[Wa61] H. Wang, "Proving Theorems by Pattern Recognition II," *Bell Systems technical Journal,* 40, 1961, pp. 1-41.

[Wa93] E. Wanke, "Paths and Cycles in Finite Periodic Graphs," *Proc. 20th Symposium on Math. Foundations of Computer Science (MFCS)*, LNCS 711, Springer-Verlag, 1993, pp. 751-760.