Recent Results on the Complexity of Problems Related to Petri Nets

Rodney R. Howell and Louis E. Rosier

Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712

Abstract

In this paper, we examine the complexity of the boundedness, containment, equivalence, and reachability problems for certain subclasses of Petri nets (PNs) (equivalently vector addition systems (VASs), vector addition systems with states (VASSs), or vector replacement systems (VRSs)). Specifically, we consider the complexity of the boundedness problem for general VASSs, fixed dimensional VASSs, and conflict-free VRSs. We consider the complexity of the remaining problems for bounded VASSs, 2-dimensional VASSs, and conflict-free VRSs. Instances in each of these classes are known to have effectively computable semilinear reachability sets (SLSs). In each case, our results are derived by showing how to obtain succinct and sometimes special representations of the associated SLSs. The results discussed here constitute a summary of results obtained elsewhere by the authors. No proofs appear in this document, although we do strive to outline the general strategies involved. Readily available sources for the detailed proofs are indicated.

1. Introduction

In this paper, we summarize results presented in [12, 13, 14, 37] concerning the complexity of various problems related to Petri nets (PNs). We deal with several formalisms, each of which is in some sense equivalent to the Petri net formalism. These formalisms include vector addition systems (VASs), vector addition systems with states (VASSs), and vector replacement systems (VRSs). Furthermore, we place certain restrictions on the various formalisms so that we can examine the complexities of a number of problems for several subclasses of Petri nets. Besides the general case, we study bounded VASs and VASSs, 2-dimensional VASSs, and conflict-free VRSs, VASs, and PNs. For each of these classes we examine the complexities of problems selected from the following: boundedness, containment, equivalence, and reachability.

We start by examining the complexity of the boundedness problem (BP) for VASs and VASSs. This problem was first considered in [21], where it was shown to be decidable. However, the algorithm presented there was basically an unbounded search, and consequently no complexity analysis was shown. Subsequently, in [26], a lower bound of $O(2^{c*m})$ space was shown, where m represents the dimension of the problem instance (and c is some constant). Finally, an upper bound of $O(2^{c*n*\log n})$ space was given in [36]. Here, however, n represents the size or number of bits in the problem instance. A close analysis of the result in

[26] reveals, in terms of n, a lower bound of $O(2^{c^*\sqrt{n}})$ space, since the size of the systems constructed in [26] required $O(m^2)$ bits. Similar bounds hold for the covering problem. See [36]. Whether the gap between the upper and lower bound could be reduced (with respect to the covering problem) was posed as an open problem in [28].

In Section 2, we illustrate how these bounds can be sharpened and examine the complexity of the BP for VASSs having a fixed dimension. A detailed presentation of these results can be found in Rosier and Yen [37]. Let VASS(k,l,n) denote the class of k-dimensional n-state VASSs, where the largest integer mentioned, in an instance, can be represented in l bits. Via a modification of the technique used by Rackoff, we illustrate that the BP[1] for VASS(k,l,n), can be solved in $O((l+\log n)*2^{c*k*\log k})$ nondeterministic space. By modifying Lipton's result, a lower bound is then shown of $O((l+\log n)*2^{c*k})$ nondeterministic space. Thus, the upper bound is optimal with respect to parameters l and n, and is nearly optimal with respect to the parameter k. This yields an improvement over the result of Rackoff, especially when compared with the lower bound of Lipton. This is because the lower bound of $O(2^{c*k})$ space was essentially given for VASS(k,1,1). Now Rackoff's corresponding upper bound, just for the instances of VASS(k,1,1) constructed by Lipton, is no better than $O(2^{c*k^2*\log k})$ space. (In general, it can get much worse.) Our result, however, yields an upper bound of $O(2^{c*k*\log k})$ over the entire class (VASS(k,1,1)). We also investigate the complexity of this problem for small, but fixed, values of k. For example, we see that the BP is PSPACE-complete for 4-dimensional VASSs, and NP-hard for 2-dimensional VASSs. The above results can then be extended for the case without states. In particular, we see that the BP is NP-hard for VASS(3,l,1) and PSPACE-complete for VASS(4,l,1). Extensions to related problems (e.g. covering and reachability) are also discussed.

The containment problem (CP), equivalence problem (EP), and reachability problem (RP) for PNs (VASs, VASSs, or VRSs) are the subject of many unanswered questions concerning computational complexity. The CP and EP are, in general, undecidable [1, 10]. However, the RP is decidable [24, 29], and, for classes of PNs (VASs, VASSs, VRSs) whose reachability sets are effectively computable semilinear sets (SLSs), so are the CP and EP. Classes whose reachability sets are effectively computable SLSs include bounded VASs [21], 5-dimensional VASs (or, equivalently, 2-dimensional VASSs) [11], conflict-free VASs [6], persistent VASs [9, 25, 30, 33], and regular VASs [8, 40]. For each of these classes, the algorithm which generates the SLS representation of the reachability set is a search procedure

[1]More precisely, the problem of deciding whether a system is unbounded.

that is guaranteed to terminate. However, no analysis of when termination will occur is provided, and thus no complexity results are obtained. The best known lower bound for the general RP is exponential space [26]. The RP for conflict-free VASs has been shown to be NP-hard [20]. The perhaps best studied class is that of symmetric VASs. For this class, the CP, EP, and RP have been shown to be exponential space complete [3, 16, 28]. The result establishing the complexity of containment and equivalence for symmetric VASs was just shown recently by Huynh in [16]. Prior to the work described here, few other complexity results appear to have been known concerning these problems.

In Sections 3-5, we examine the complexity of the CP, EP, and RP for three classes of PNs. Specifically, we consider the complexity of these problems for bounded VASSs, 2-dimensional VASSs, and conflict-free VRSs (VASs, PNs). Instances in each of these classes are known to have effectively computable semilinear reachability sets. By giving upper bounds on the sizes of the SLS representations, we achieve upper bounds on each of the aforementioned problems. The results discussed here are essentially a summary of the results presented by Howell, Huynh, Rosier, and Yen in [12] and Howell and Rosier in [13].

In Section 3, we concern ourselves with examining the complexity of the CP and EP for bounded VASSs. A detailed description of the results reported in this section can be found in Howell, Huynh, Rosier, and Yen [12]. In the case of bounded VASSs, the SLS representation is simply a listing of the reachability set; therefore, we derive a bound on the norm of any reachable vector based on the dimension, number of states, and amount of increment caused by any move in the VASS. The bound we derive shows an improvement of two levels in the primitive recursive hierarchy over results previously obtained by McAloon [31], thus answering a question posed by Clote [5]. We then show this bound (on the norm of any reachable vector) to be optimal. As a consequence of our analysis, we also derive similar bounds for bounded VASs.

In Section 4, we describe results concerning the complexity of the CP, EP, and RP for 2-dimensional VASSs. (Recall that the BP for 2-dimensional VASSs was considered in Section 2.) A detailed description of the results reported in this section can be found in Howell, Huynh, Rosier, and Yen [12]. The results are obtained in part via an analysis of an algorithm given by Hopcroft and Pansiot [11] that generates a semilinear set (SLS) representation of the reachability set. Specifically, [12] shows that the algorithm operates in $2^{2^{c*l*n}}$ nondeterministic time, where l is the length of the binary representation of the largest integer in the VASS, n is the number of transitions, and c is some fixed constant. Examples are given in [12] for which this algorithm will take $2^{2^{d*l*n}}$ nondeterministic time for some positive constant d. Finally, we can modify the algorithm to be deterministic in such a way that it still requires no more than $2^{2^{c*l*n}}$ time. From this upper bound and special properties of the generated SLSs, we derive upper bounds of DTIME($2^{2^{c*l*n}}$) for the three problems mentioned above.

In Section 5, we give completeness results for the BP, CP, EP, and RP for conflict-free VRSs (VASs, PNs). For the BP, we give an $O(n^{1.5})$ upper bound, from which we can show the problem to be PTIME-complete. This result first appeared in Howell, Rosier, and Yen [14]. The $O(n^{1.5})$ upper bound represents an improvement over the previously best-known upper bound of exponential time shown by Landweber and Robertson [25]. We then give an NP upper bound for the RP. This result represents the first primitive recursive upper bound known for the problem. Since Jones, Landweber, and Lien [20] have shown this problem to be NP-hard, it follows that the problem is NP-complete. From these results, we can show that the CP and EP are Π_2^P-complete, where Π_2^P is the set of all languages whose complements are in the second level of the polynomial-time hierarchy. In showing the upper bound, we first show that the reachability set has a SLS representation that is exponential in the size of the problem description, but which has a high degree of symmetry. We are then able to modify a proof given by Huynh (concerning SLSs) to complete our upper bound proof. All of our results for conflict-free VRSs also hold for conflict-free VASs and are polynomially related to our bounds for conflict-free PNs. A more detailed discussion of the results for containment, equivalence, and reachability for conflict-free VRSs can be found in Howell and Rosier [13].

Preliminary Definitions

Let Z (N, N^+, R) denote the set of integers (nonnegative integers, positive integers, rational numbers, respectively), and let Z^k (N^k, R^k) be the set of vectors of k integers (nonnegative integers, rational numbers). For a vector $v \in Z^k$, let v(i), $1 \leq i \leq k$, denote the i-th component of v. Let $Z^{k \times m}$ ($N^{k \times m}$, $R^{k \times m}$) be the set of k×m matrices of integers (nonnegative integers, rational numbers). For a matrix $V \in Z^{k \times m}$, let V(i,j), $1 \leq i \leq k$, $1 \leq j \leq m$, denote the element in the ith row and jth column of V, and let v_j denote the jth column of V. For a given value of k, let **0** in Z^k denote the vector of k zeros (i.e., **0**(i)=0 for i=1, . . . ,k). For any $v \in Z^k$, we define the *norm* of v, $||v||$, as $\sum_{i=1}^{k} |v(i)|$. (Note that this is often called the *1-norm.*) Now given vectors u, v, and w in Z^k we say:

- v=w iff v(i)=w(i) for i=1, . . . ,k;

- $v \geq w$ iff $v(i) \geq w(i)$ for $i=1, \ldots, k$;
- $v > w$ iff $v \geq w$ and $v \neq w$;
- and $u=v+w$ iff $u(i)=v(i)+w(i)$ for $i=1, \ldots, k$.

A $k \times m$ *vector replacement system* (VRS), is a triple (v_0,U,V), where $v_0 \in N^k$, $U \in N^{k \times m}$, and $V \in Z^{k \times m}$, such that for any i,j, $1 \leq i \leq k$, $1 \leq j \leq m$, $U(i,j)+V(i,j) \geq 0$. v_0 is known as the *start vector*, U is known as the *check matrix*, and V is known as the *addition matrix*. A column u_j of U is called a *check vector*, and a column v_j of V is called an *addition rule*. For any $x \in N^k$, we say addition rule v_j is *enabled* at x iff $x \geq u_j$. A sequence $<y_1,\ldots,y_n>$ of rules in V is *enabled* at a vector x iff for each j, $1 \leq j \leq n$, y_j is enabled at $x+y_1+ \cdots +y_{j-1}$. The *reachability set* of the VRS $\mathcal{V}=(v_0,U,V)$, denoted by $R(v_0,U,V)$ (or $R(\mathcal{V})$), is the set of all vectors z, such that $z=v_0+y_1+ \cdots +y_n$ for some $n \geq 0$, where each y_j $(1 \leq j \leq n)$ is a column of V, and $<y_1,\ldots,y_n>$ is enabled at v_0. If the sequence of rules $<y_0,\ldots,y_n>$ is enabled at v_0, then we say that the sequence $<w_0,\ldots,w_n>$, where $w_j=v_0+\sum_{i=1}^{j} y_i$, $1 \leq j \leq n$, is a *path* in (v_0,U,V). If along the path there exist r and s, $1 \leq r<s \leq n$, such that $w_r \leq w_s$ $(w_r<w_s)$, then we say that $\pi=<w_r,\ldots,w_s>$ is a *loop* (*positive loop*), and that π is *enabled* at w_{r-1}. Let Ψ denote the *Parikh mapping*, such that if θ is a sequence of rules in V, then $\Psi(\theta) \in N^m$, and $\Psi(\theta)(j)$ is the number of occurrences of v_j in θ. Let $\Delta(\theta)$ denote the displacement of θ.

For a given $k \times m$ matrix V, the *minimal check matrix* is a $k \times m$ matrix U in which $U(i,j) = \max(0, -V(i,j))$. If U is the minimal check matrix for V, we abbreviate (v_0,U,V) as (v_0,V), and call (v_0,V) a *vector addition system*. We are sometimes interested in the set of all VRSs (VASs) having k rows (i.e., we are not interested in the number of columns). We will refer to this set as the set of k-dimensional VRSs (VASs, respectively). A k-dimensional *vector addition system with states* (VASS) is a 5-tuple (v_0,V,p_0,S,δ) where v_0 is the same as defined above, V is a finite set of addition rules, S is a finite set of states, δ $(\subseteq S \times S \times V)$ is the transition relation, and p_0 is the initial state. Elements (p,q,x) of δ are called *transitions* and are usually written $p \rightarrow (q,x)$. A *configuration* of the VASS is a pair (p,x), where p is in S and x is a vector in N^k. (p_0,v_0) is the initial configuration. The transition $p \rightarrow (q,x)$ can be applied to the configuration (p,v) and yields the configuration (q,v+x), provided that $v+x \geq \mathbf{0}$. In this case, (q,v+x) is said to *follow* (p,v). Let σ_0 and σ_t be two configurations. Then σ_t is said to be *reachable* from σ_0 iff $\sigma_0=\sigma_t$ or there exist configurations $\sigma_1, \ldots, \sigma_{t-1}$ such that σ_{r+1} follows σ_r for $r=0, \ldots, t-1$. We then say $\sigma=<\sigma_0, \ldots, \sigma_t>$ is a *path* in (v_0,V,p_0,S,δ). The *reachability set* of the VASS $\mathcal{V}=(v_0,V,p_0,S,\delta)$, denoted by $R(v_0,V,p_0,S,\delta)$

(or $R(\mathcal{V})$), is the subset of $S \times N^k$ containing all configurations reachable from (p_0,v_0). It is easily seen that these definitions for VASs, VASSs, and VRSs are equivalent to those given in [21, 11, 23].

The *reachability problem* (RP) for VASs (VASSs, VRSs) is to determine, for a given VAS (VASS, VRS) $\mathcal{V}$ and a vector v, whether $v \in R(\mathcal{V})$. The *containment* and *equivalence problems* (CP, EP, respectively) are to determine, for two given VASs (VASSs, VRSs) $\mathcal{V}$ and $\mathcal{V}'$, whether $R(\mathcal{V}) \subseteq R(\mathcal{V}')$ and whether $R(\mathcal{V}) = R(\mathcal{V}')$, respectively. A VAS (VASS, VRS) $\mathcal{V}$ is said to be *bounded* iff for each row i, there is a constant c such that if $v \in R(\mathcal{V})$, then $v(i) < c$. The *boundedness problem* (BP) for VASs (VASSs, VRSs) is the problem of determining whether a given VAS (VASS, VRS) is bounded.

Part of our exposition involves semilinear sets; hence, we define here the terms used in this paper. For any vector $v_0 \in N^k$ and any finite set $P(=\{v_1,\ldots,v_m\}) \subseteq N^k$, the set $\mathcal{L}(v_0,P) = \{x : \exists k_1,\ldots,k_m \in N \text{ and } x = v_0 + \sum_{i=1}^{m} k_i v_i\}$ is called the *linear set* with *base* v_0 over the set of *periods* P. The *size* of the linear set $\mathcal{L}(v_0,P)$, denoted by $|\mathcal{L}(v_0,P)|$, is defined to be $\sum_{i=0}^{m} k^* \log_2 ||v_i||$. (I.e., the number of bits needed to represent the linear set.) A finite union of linear sets is called a *semilinear set* (SLS for short).

2. The Complexity of the General BP

In this section, we examine the complexity of the BP for VASSs with respect to the following three natural numeric parameters of an instance:

- the dimension,
- the maximum size of any integer mentioned in the description, and
- the number of states.

For ease of expression, we let VASS(k,l,n) denote the class of k-dimensional n-state VASSs where the largest integer mentioned, in an instance, can be represented in l bits. (Note that this definition agrees with the one in [37], but differs slightly from the one in [12].) In Subsection 2.1, we illustrate via a modification of Rackoff's technique that the BP, for VASS(k,l,n), can be solved in $O((l+\log n)^* 2^{c^* k^* \log k})$ nondeterministic space. (All of our results in this section are expressed with respect to nondeterministic complexity classes unless otherwise stated. This could also be said for [26] and [36], although in their case such algorithms can be made deterministic without altering the complexity class, as only the constant c gets changed [38]. Our subsequent discussion, however, involves subexponential complexity classes where this is not necessarily the case. Hence, it is important in our analysis that we consider the problem of deciding whether systems are unbounded rather than bounded.) This offers an improvement over the result in [36], especially when compared with the best known lower bound. This is because the lower bound of $O(2^{c*k})$ space was essentially given for VASS(k,1,1). Now the corresponding upper bound from [36], just for the instances of VASS(k,1,1) considered in [26], is no better than $O(2^{c*k^2*\log k})$ space. (In general, it can get much worse.) Our result, however, yields an upper bound of $O(2^{c*k*\log k})$, over the entire class.

Whenever k is a fixed known constant, however, our algorithm requires at most nondeterministic linear space in terms of l. In Subsection 2.2, we see that the BP is PSPACE-complete for 4-dimensional VASSs. Although we are unable to provide the corresponding result for either 2 or 3-dimensional VASSs, we are able to show that the problem is NP-hard for these classes. Similar results hold for the case without states. In particular, we see that the BP is NP-hard for VASS(3,l,1) and PSPACE-complete for VASS(4,l,1). Consequently, it is unlikely that the problem can be solved in PTIME with respect to the parameter l. When k=1, however, the BP can be solved in PTIME. When both k and l are considered to be fixed known constants the problem becomes NLOGSPACE-complete. Hence, the complexity of the algorithm given in Subsection 2.1 is unlikely to be improved with respect to the parameter n either.

Now in Subsection 2.3, we illustrate how the proof of [26] can be augmented to obtain a lower bound of $O((l+\log n)*2^{c*k})$ nondeterministic space for VASS(k,l,n). In order to do this, we define a class of problems each of which has three natural numeric parameters, say k, l and n, whose nondeterministic space complexity is bounded by $2^{k}*(l+\log n)$. We then establish that the BP for VASS(k,l,n) is as "hard" as any problem in this class with respect to a certain special type of reducibility. Our goal here is to provide a simultaneous lower bound over the three parameters, as opposed to examining the lower bound when one (or two) of the parameters is fixed. (Thus, an algorithm for this problem with nondeterministic space complexity $O(2^{c*k}+l+\log n)$ cannot exist. Note, however, that the existence of such an algorithm was not precluded by the results given in the previous subsection.) Thus, the algorithm presented in Subsection 2.1, is optimal with respect to parameters l and n, and is nearly optimal with respect to the parameter k. We do not know at this time, however, whether the log k factor can be eliminated.

Before continuing to the discussion of our results, let us briefly mention that these results can be extended to some other related problems concerning VASSs. For example, the lower bounds established in Subsection 2.3 hold for the covering and reachability problems.

Using a similar approach, as was used in Subsection 2.1, an upper bound of $O((l+\log n) * 2^{c*k*\log k})$ can be shown for the covering problem. (Finding better upper/lower bounds for this problem was mentioned as an open problem in [28].) Previously, the best known bounds for this problem were essentially the same as those for the BP. See [26, 28, 36].

2.1. The Upper Bound

The BP for VASs was first studied in [21], where a nonprimitive recursive decision procedure was proposed. This procedure is based on a search algorithm which generates the reachability set $R(v_0,V)$, and at the same time, attempts to find a "pumpable loop", that can be exploited to reach an arbitrary number of distinct vectors. It was shown that either the reachability set is finite or such a loop exists. Hence, the algorithm must eventually terminate. Unfortunately, the size of $R(v_0,V)$ is not bounded by a *primitive* recursive function of the size of (v_0,V). Rackoff, in [36], subsequently showed that if such a pumpable loop could be executed by some computation of the system, then it could be executed by a "short" computation. As a result, Rackoff presented an algorithm which required at most $2^{c*m*\log m}$ space, for some constant c. Here, the parameter m represents the size (i.e., number of bits) of the instance. Actually, this parameter, upon careful examination, can be decomposed into three natural parameters k, l and n, where k and n are the dimensions of the addition matrix V, and l is the maximum length of the binary representation of each component in V. (The reader should note that these parameters are equally natural for PNs. There k would represent the number of places, l the maximum number of inputs or outputs of a transition and n the number of transitions. See e.g. [35].) However, we can assume without loss of generality that $n \leq (2*2^l+1)^k$. Hence, for simplicity, we use VAS(k,l) to denote the class of VASs with parameters k and l. Later, when we discuss VASSs, we introduce another parameter m to represent the number of states. It should be clear that the three parameters k, l, and m are mutually independent. Therefore, the class of VASSs with parameters k, l and m will be denoted as VASS(k,l,m).

In the remainder of this subsection, we sketch how one can obtain an upper bound of $O((l+\log n)*2^{c*k*\log k})$ nondeterministic space, where c is a constant and n is the number of rules, for the BP for VAS(k,l). What we actually show, as did Rackoff, is a bound on the length of the shortest system computation, if one exists, which will execute a pumpable loop. The general strategy is, in fact, a modification of the one in [36]. Formal details can be found in [37].

We require the following definitions. Let $w \in Z^k$ and $0 \leq i \leq k$. The vector w is *i-bounded* if $w(j) \geq 0$ for $1 \leq j \leq i$. Let $p = w_0,\ldots,w_m$ be a sequence of vectors, we say p is i-bounded if every member in p is i-bounded. Moreover, if $w_j < w_m$ for some j, $0 \leq j < m$, then p is said to be *self-covering*. Let $0 \leq i \leq k$, and let V be an arbitrary addition matrix. In what follows, we consider a generalization of the notion of a path. Let $\theta = z_1,\ldots,z_m$ be any sequence of addition rules in V. We then call $p = w_0,\ldots,w_m$, where $w_j = v + \sum_{h=1}^{j} z_h$, a *g-path* in (v,V). Note that a k-bounded g-path is a path. For each $v \in Z^k$, define $m'(i,v)$ to be the length of the shortest i-bounded, self-covering g-path in (v,V), if one exists; otherwise, define $m'(i,v)=0$. Define $g(i) = \max\{m'(i,v) \mid v \in Z^k\}$. This function g will then give us a bound on the length of the shortest computation which can execute a pumpable loop. Note that this upper bound does not depend on any specific start vector.

In [37], we illustrate that:

(1) $g(0) < (2^l n)^{ck}$, for some constant c independent of l, k and n, and

(2) $g(i+1) < (2^{2l} * g(i))^{k^c}$, for some constant c independent of l, k and n.

Now by recursively applying (1) and (2), it is easy to show that $g(k) \leq (2^l * n)^{2^{c*k*\log k}}$. This means if a VAS is unbounded, there must exist a path, of length no more than $(2^l * n)^{2^{c*k*\log k}}$, that leads to a "pumpable loop". Therefore, we have:

Theorem 2.1: The BP for VAS(k,l) can be decided in $O(2^{c*k*\log k} * (l + \log n))$ nondeterministic space, for some constant c independent of l, k and n.

Now consider a VASS in the class VASS(k,l,m). Clearly n, the number of possible rules, is no more than $m * (2 * 2^l + 1)^k$. As a result we have:

Corollary 2.2: The BP for the class VASS(k,l,m) can be solved in $O((l + \log m) * 2^{c*k*\log k})$ nondeterministic space, for some constant c independent of l, k and m.

2.2. The Complexity of the BP When k Is a Fixed Known Constant

In this subsection, we focus our attention on the BP for VASS(k,l,m) when k is fixed. A summary of these results can be found in Table 2.3. In [37] we show that the BP is PSPACE-complete for $k \geq 4$. Consequently, it is unlikely that one can improve the complexity of the aforementioned algorithm with respect to the parameter l. At this time, we do not know whether the PSPACE-complete result can be improved for the case when k=2 or 3.[2] However, in [37], we show that it is NP-hard when k=2. Now, when k=1, there

[2]When k=3 it has recently been claimed that the problem is PSPACE-complete [4].

exists a PTIME algorithm. If l is also fixed, then the problem becomes NLOGSPACE-complete. However, we do not know whether the previous PTIME result can be improved to NLOGSPACE. These results should be compared with earlier PSPACE-hard results concerning VASs (or VASSs) in the literature [20], where the parameter k instead of l is allowed to vary.

	VASS(k,l,m)	VAS(k,l)
PSPACE-complete	$k \geq 4$	$k \geq 4$
NP-hard	$k \geq 2$	$k \geq 3$
PTIME	$k=1$	-----
trivial	-----	$k=1$

Table 2.3 The complexity of the BP for VASSs (VASs) where k is fixed.

2.3. The Lower Bound

In this subsection, we fix the alphabet over which problems can be specified. Without loss of generality let this alphabet be $\Sigma=\{0,1\}$. A decision problem over Σ is said to be solvable in $\psi(x)$ deterministic time ($\phi(x)$ nondeterministic space) iff there exists a deterministic (nondeterministic) Turing machine with tape alphabet Σ which decides the problem using at most $\psi(x)$ time ($\phi(x)$ space), where x is the input string. (Hereafter, deterministic (nondeterministic) Turing machines are abbreviated as DTMs (TMs).) Note that ψ (ϕ) need not be a function of the input length. A function g: $\Sigma^* \rightarrow \Sigma^*$ is said to be computable in S(m) space iff there exists a DTM M such that, given an input $x \in \Sigma^*$, M will output g(x) using at most S(|x|) space. Now, consider two problems L and L′ over Σ^*. L is said to be (S,Q)-reducible to L′ via the function g iff g is computable in S(|x|) space such that:

- $x \in L$ iff $g(x) \in L'$, and
- $\forall x \in \Sigma^*$, $|g(x)| \leq Q(|x|)$.

From [22], we obtain (1) and (2) below.

(1) If a function f: $\Sigma^* \rightarrow \Sigma^*$ is computable by an S(n) space bounded DTM M with tape symbols {0,1,#} such that at any time the worktape contains at most k #s, then f is computable in S(n) + (k+2)*log S(n) space by a DTM M′ with tape symbols {0,1}.

(2) Suppose L is (S,Q)-reducible to L′ via the function g.

(a) If L′ is solvable in $\psi(x)$ deterministic time, then there is a constant c such that L is solvable in $\psi(g(x))+c*|x|*S(|x|)*2^{S(|x|)}$ deterministic time.

(b) If L′ is solvable in $\phi(x)$ nondeterministic space, then there is a constant c such that L is solvable in $\phi''(x)+c*\log\phi''(x)$ nondeterministic space, where

$\phi''(x)=\phi(g(x))+2*\log(Q(|x|))+S(|x|)$.

In [37], we establish a lower bound for the VASS BP in terms of the three natural numeric parameters, k, l and n. The goal is to provide a simultaneous lower bound over the three parameters, as opposed to examining the lower bound when one (or two) of the parameters is fixed. In order to do this, we focus on the following class:

C: A problem P belongs to C if P is a problem with three natural numeric parameters, say k, l and n, that can be solved in $2^{k}*(l+\log n)$ nondeterministic space, and for any instance x, of P, the following three conditions are satisfied:

1. $|x| \geq \max\{k,l,n\}$, ($|x|$ represents the length of x),
2. k, l and n can be computed from x and written down in deterministic log $|x|$ space, <u>and</u>
3. $k\geq\log|x|$ or $l\geq\log|x|$ or $n\geq|x|/2$.

Let L and L′ be two problems with natural parameters (k,l,n) and (k',l',n'), respectively. Let d be a constant. For such problems we say L is d_(S,Q)-reducible to L′ iff there exists a function g such that L is (S,Q)-reducible to L′ via g and $k'\leq d*k$, $l'\leq d*l$, and $n'\leq d*n^2$. (Here for an arbitrary $x\in\Sigma^*$, k, l and n (k', l' and n', respectively) denote the natural parameters with respect to x and L (g(x) and L′, respectively).) Let $\mathcal{C}$ be a class of problems over Σ. L is said to be *$\mathcal{C}$-hard* with respect to d_(S,Q)-reducibility if for every L′ in $\mathcal{C}$, there exists a constant c such that L′ is d_(S(n),c*Q(n)) reducible to L. In [37] we show that the BP for the class VASS(k,l,n) is hard for C, with respect to d_$((2+\epsilon)\log(\log|x|+\log n), |x|^3)$-reducibility for some constant d and all $\epsilon>0$.

To establish this conclusion, the following intermediate result is required. The proof is similar to the one in [26], where Lipton constructed a VAS to simulate a multiple counter machine. A rather lengthy sketch is provided in [37].

(3) There exists a positive integer constant h, such that for any 3-tuple of integers k, l and n one can construct a VASS in VASS(h*k,l,n), that can manipulate a counter, whose value can range from 0 to $(n*2^{l})^{2^k}$. Furthermore, this counter can be incremented, decremented and tested for zero. (I.e., the resulting VASS can simulate a counter machine whose counter is bounded by $(n*2^{l})^{2^k}$.)

Now, using (3), one can construct a VASS in VASS(h*k,l,n) (for some constant h) such that a pair of positions (r,r′) can be used to simulate a counter. By using no more than three times the number of positions one can then construct such a VASS that can simulate, in some sense, a three counter machine (3CM), ala Minsky [32]. Now in [37] we show how an arbitrary problem in C can be reduced to the BP for VASS(k,l,n). As a result, we obtain the following theorem and associated corollary:

Theorem 2.4: The BP for VASS(k,l,n) is hard for C, with respect to d_((2+ϵ)(log|x|+log n),|x|^3)-reducibility.

Corollary 2.5: There exist some constants c, c′ and h independent of k, l and n, such that the BP for VASS(k,l,n) requires $2^{k/c\text{-}c'}*(l+\log n)$ nondeterministic space for $k \geq h$.

Note that given a DLBA, we could construct an equivalent 3CM in which two counters are used to simulate the tape and the third is used as a working counter. Hence, one can construct a VASS in VASS(6,c*|x|,n) to simulate the 3CM in such a way that each counter is represented by a pair of positions. Consequently, for $k \geq 6$ the BP becomes PSPACE-hard. The reader should recall, however, that Subsection 2.2 illustrated that the BP was PSPACE-hard for $k \geq 4$.

3. The Complexity of Containment and Equivalence for Bounded VASSs

In this section, we describe results from [12] concerning the complexity of the CP and EP for bounded VASSs. Recently, Mayr and Meyer [27] showed that the CP and EP for bounded VASs are not primitive recursive. Subsequently, McAloon [31] showed that the problems are primitive recursive in the Ackermann function, and Clote [5] showed the finite CP[3] (FCP) to be DTIME(Ackermann) complete. Let $f_1(x)=2x$ and $f_n(x)=f_{n-1}^{(x)}(1)$ for $n>1$, where $f_i^{(m)}$ is the m-th fold composition of f_i. Using a combinatorial argument, McAloon showed an upper bound for the time complexity of the FCP that can be shown to be at least $f_{k+1}(m)$, where k is the row dimension and m is the maximum sum of the elements of any vector in the VAS (see also [5]). Clote [5] subsequently used Ramsey theory to give an upper bound of approximately $f_{k+6}(m)$ and posed a question as to whether McAloon's bound could be improved. It follows that these bounds also hold for the size of bounded VASs. McAloon's bound on the size of bounded VASs is close to optimal. See [30, 27, 34, 40].

Let BV(k,b,n) be the class of k-dimensional n-state bounded VASSs where the maximum increase in the norm of a vector (i.e., the sum of the absolute values of its elements) caused by any move is b. (Assume the start vector is **0**.) In [12], we use a tree construction technique to derive an upper bound on the largest norm of any vector reachable in BV(k,b,n). The bound we derive for k-dimensional VASs is $f_{k-1}(d*m^2)$, $k \geq 2$, ($f_{k-1}(d*m)$ for

[3]I.e., the CP for bounded systems.

$k \geq 4$), where m is the maximum sum of the elements of any vector in the VAS, and d is a constant. By then considering the addition of states and the restriction of the start vector to **0**, we derive a bound of $f_k(c^*\max(n,\log b))$ on the norm of the largest vector reachable in BV(k,b,n), where $k \geq 3$ and c is a constant. Furthermore, we show that this bound is tight for b=1. (I.e., we illustrate for each k and m a VASS in BV(k,1,m*(2*k-1)+2) that can generate a vector with norm $f_k(m)$.) These results immediately yield, for the k-dimensional VAS FCP, a bound of $f_{k-1}(d'^*m)$ time, for $k \geq 4$ and some constant d'. This bound represents an improvement of two levels in the primitive recursive hierarchy over McAloon's result, thus answering the question posed by Clote. Since we do not know of any attempts to use tree construction techniques similar to ours in analyzing combinatorial problems, and because our techniques yield better results than the standard combinatorial techniques applied in the past to this problem, we surmise that our techniques may have other applications. Finally, we show that the CP and EP (for BV(k,b,n)) require at least time $f_{k-c}(d^*n)$ infinitely often for some constants c and d. The proof is such that each position in the constructed VASS can be bounded by $f_{k-c}(d^*n)$. Hence, if we considered the entire class of VASSs whose positions were bounded by $f_k(n)$ (rather than just BV(k,b,n)) our lower bound would be tight. We surmise, therefore, that the constant c can be eliminated.

In this section, we will assume that the start vector for a VASS is always **0**. (Note that $R(v_0,V,p_0,S,\delta)=R(\mathbf{0},V\cup v_0,q,S\cup\{q\},\delta')$ for some $q \notin S$ and some δ'.) Let BV(k,b,n) be the set of all VASSs $(\mathbf{0},V,p_0,S,\delta)$ such that $R(\mathbf{0},V,p_0,S,\delta)$ is finite, $V \subseteq Z^k$, $|S|=n$, and $\max\{\sum_{i=1}^{k} v(i) : v \in V\}=b$. We define $\mu(k,b,n)$ as the maximum norm of any vector reachable by a VASS in BV(k,b,n). Let σ be a path in a VASS. We define the *monotone increasing component* of σ, $\iota(\sigma)$, to be the sequence of configurations σ_i in σ for which all previous configurations in σ having the same state as σ_i have a vector with strictly smaller norm than that of σ_i. If σ is a path in a VASS in BV(k,b,n), then $\iota(\sigma)$ clearly has finite length.

3.1. Bounds on the Sizes of Bounded VASSs

The general idea in [12] is to arrange the monotone increasing component of a path in a VASS into a tree in which any proper subtree contains only configurations whose states are the same and whose vectors have identical values in certain positions. In particular, in a subtree rooted at depth i (where the root of the tree is defined to be at depth 0), $i \geq 1$, all vectors will agree in at least i-1 positions. The resulting tree has certain properties which allow us to give a tight upper bound on its size, and hence, on the length of the monotone increasing component. We define $\mathcal{T}(k,b,n)$ as the set of trees T having the following properties:

1. T has height $\leq$ k (i.e., the longest path from the root to a leaf is no more than k);

2. The root node of T is labelled 0 and has no more than n-1 children;

3. The nodes in T have integer labels such that for any node labelled $r>b$, there is a node labelled s, $r-b \leq s<r$;

4. The label of any node in T is less than the label of any of its children;

5. The number of children of any node of depth i, $1 \leq i \leq k-1$, is no more than the node's label.

Each node in a tree $T \in \mathcal{T}(k,b,n)$ will represent a configuration in $\iota(\sigma)$; the node label will be the norm of the vector in that configuration. Each proper subtree of T will represent a hyperplane having dimension k-j+1, where j is the depth of the root of the subtree. The root of T represents the initial configuration. The remainder of T is divided into subtrees rooted at depth 1, each representing the set of configurations in $\iota(\sigma)$ having some particular state. Suppose some state q is entered for the first time with vector v. Then the configuration (q,v) is represented by a node with depth 1 in T. Now all subsequent configurations in $\iota(\sigma)$ containing q must have a vector v′ such that $v'(i)<v(i)$ for some i; otherwise, the path from (q,v) to (q,v′) could be pumped, producing infinitely many configurations. Consider the hyperplane of dimension k-1 in which all vectors w are such that $w(i)=v'(i)$. For an arbitrary v, there are at most $||v||$ such hyperplanes, each defined by the values of i and $v'(i)<v(i)$. The subtrees rooted at depth 2 in T represent these hyperplanes, their roots representing the first configuration occurring in that hyperplane. Likewise, the subtrees rooted at depth j represent hyperplanes of dimension k-j+1. It is easily seen that T satisfies properties 1, 2, 4, and 5 above; furthermore, since b is the maximum increase in norm caused by any transition, property 3 must also hold. As a result, we have:

(1) Let σ be a path in a VASS in BV(k,b,n), $\iota(\sigma)=<\sigma_0, \ldots, \sigma_t>$. There is a tree $T \in \mathcal{T}(k,b,n)$ with t+1 nodes whose labels are the norms of the vectors in $\iota(\sigma)$.

The following facts are established in [12]:

(2) For any tree $T \in \mathcal{T}(k,b,n)$, there is a tree $T' \in \mathcal{T}(k,b,n)$ with the same number of nodes as T such that the labels on all nodes of any given depth are nondecreasing from left to right.

(3) For any k, b, and n, $\mathcal{T}(k,b,n)$ contains a tree of maximal size (i.e., a tree having as many nodes as any other tree in $\mathcal{T}(k,b,n)$).

(4) Any tree in $\mathcal{T}(k,b,n)$ having maximal size has its node labels arranged in order of a depth-first (preorder) traversal.

(5) Let S(k,b,n,i,x) be the set of subtrees in $\mathcal{T}$(k,b,n) whose root is at depth i with label x. The largest element of S(k,b,n,i,x) has its node labels arranged in order of a depth-first (preorder) traversal.

(2) can be shown by observing that if a tree $T \in \mathcal{T}(k,b,n)$ has its top j levels ordered, then any subtrees rooted at the next level whose whose roots are out of order may be swapped, resulting in a tree $T' \in \mathcal{T}(k,b,n)$. (3) is shown by induction on k. The idea is to show that if the lemma were false for k but true for k-1, then any tree in $\mathcal{T}(k,b,n_0)$, where n_0 is the smallest integer such that $\mathcal{T}(k,b,n_0)$ has unbounded tree sizes, can be rearranged to form a tree in $\mathcal{T}(k\text{-}1,b,n_0+u)$ for some u. Finally, (4) is shown by rearranging a tree $T \in \mathcal{T}(k,b,n)$ whose node labels are not in order of a depth-first traversal into a tree T′ that has room for at least one more node. After we rearrange T so that each level has nondecreasing node labels from left to right, we wish to move the smallest node label s which does not appear in depth-first order into its proper place in a depth-first ordering. It is not hard to show that by repeatedly moving the leftmost descendants of s on successive levels upward to replace their parents, we can make room for the node displaced by s and any extra subtrees beneath this node. These subtrees are all moved upward in the tree, so there is room for new nodes at the bottoms of these subtrees. (5) follows as a corollary to (4).

Since the largest tree T in $\mathcal{T}$(k,b,n) has its node labels arranged in depth-first order, the value of any given node in T (except the root) and the number of children it has (if its depth is k-1 or less) is simply b plus the largest label in the subtree rooted at its left-hand sibling, or b plus the label of its parent if it has no left-hand sibling. It is therefore a straightforward matter to derive a recurrence relation for the largest node label in T. If we let $\lambda_1(b,n)=n*b$, $\lambda_2(b,n)=n*\max(\log\ b,1)$, and $\lambda_i(b,n)=\max(n,\log\ b)$ for $i \geq 3$, we can easily show the following theorems:

Theorem 3.1: There exist constants c and d (independent of k and b) such that for any k-dimensional bounded VAS (v_0,V) with $\max\{\sum_{i=1}^{k} v(i) : v \in \{v_0\} \cup V\} = b$, $k \geq 2$, we have $\forall v \in R(v_0,V)$, $||v|| \leq f_{k-1}(c*\lambda_{k-1}(b,||v_0||))$.

Theorem 3.2: There exists a constant c (independent of k, b, and n) such that $\mu(k,b,n) \leq f_k(c*\lambda_k(b,n))$.

[12] also illustrates a VASS in BV(k,1,m*(2*k-1)+2) (and in VASS(k,1,m*(2*k-1)+2)) that can produce a vector with norm $f_k(m)$; hence, the derived bound for $\mu(k,b,n)$ is tight. This construction is similar to one given by Müller in [34], which gives a weak PN computer for Ackermann functions; in fact, a simple modification can be made to the construction in

[34] to yield the same lower bound for $\mu(k,b,n)$.

3.2. The Finite Containment and Equivalence Problems[4]

In this subsection we concern ourselves with the complexity of the CP and EP for bounded VASSs. If u is an upper bound on the norm of any vector reachable by a k-dimensional VASS (or VAS), clearly u^k is an upper bound on the number of vectors in the reachability set. It then follows that the FCP and FEP can be solved in time $O(u^{2k})$. Thus, from Theorems 3.1 and 3.2 we have the following result, which represents an improvement of two levels in the primitive recursive hierarchy over the bound provided by [31].

Theorem 3.3: There exists a positive constant c (independent of k, b, and n) such that the CP and EP can be solved in time

1. $f_k(c*\lambda_k(b,n))$ for BV(k,b,n), $k \geq 2$;
2. $f_1^4(c*n*b)$ for 2-dimensional bounded VASs whose vectors cause increments of no more than b and whose start vectors have norm n;
3. $f_{k-1}(c*\lambda_{k-1}(b,n))$ for k-dimensional bounded VASs, $k \geq 3$, whose vectors cause increments of no more than b and whose start vectors have norm n.

We can also show, by a refinement of the proof in [27], the following:

Theorem 3.4: There exist positive constants a, b and c (independent of k and n) such that the CP and EP for BV(k,1,n) require $f_{k-a}(b*n)$ time infinitely often whenever $k>c$.

In the proof of the previous theorem (see [12]), the construction is such that each position is bounded by $f_{k-a}(b*n)$. By letting $\mathbf{V}_i(n)$ denote the set of bounded VASSs whose reachability sets are bounded by $f_i(n)$, we have the following corollary:

Corollary 3.5: There exist positive constants c and d (independent of i and n) such that the time complexity of the CP and EP for $\mathbf{V}_i(n)$ are bounded above (below) by $f_i(d*n)$ $(f_i(c*n))$.

4. The Complexity of Containment, Equivalence, and Reachability for 2-dimensional VASSs

In this section, we describe results from [12] concerning the complexity of the CP, EP, and RP for 2-dimensional VASSs. The study of VASSs having a specific fixed dimension is interesting because it can be shown, using techniques similar to those in [1, 7, 10], that there

[4]As was the case with the finite containment problem (FCP) the finite equivalence problem (FEP) refers to the EP for bounded systems.

exists a k such that the CP and EP are undecidable for k-dimensional VASSs. Although the best upper bound for k is unknown at this writing, the CP and EP are known to be decidable for k=2 [11]. 2-dimensional VASSs are of additional interest because the reachability set is not, in general, semilinear when the dimension is greater than two [11]. In this section, we utilize the ideas inherent in the previous section to provide an analysis of the algorithm given in [11], which generates from an arbitrary 2-dimensional VASS the SLS representation of its reachability set. As a result of the analysis, we obtain upper bounds for the CP, EP, and RP in the case of 2 dimensions. For a given VASS, let l denote the maximum number of bits needed to represent any integer in the system, and let n denote the maximum of the number of states and the number of transitions. Specifically, we show that the algorithm of Hopcroft and Pansiot [11] operates on any 2-dimensional VASS in NTIME($2^{2^{c*l*n}}$) for some constant c. Furthermore, there are instances that require $2^{2^{d*l*n}}$ steps for some positive constant d; hence our analysis (of the algorithm) is tight. We then give a minor modification to the algorithm that reduces its complexity to DTIME($2^{2^{c'*l*n}}$) for some constant c′. The SLS constructed by the resulting algorithm contains O($2^{2^{c'*l*n}}$) linear sets. Each of these linear sets has a base with norm O($2^{2^{c'*l*n}}$) and O(2^n) periods with norm O($2^{d'*l*n}$) for some constant d′. From these properties we derive an upper bound of DTIME($2^{2^{c*l*n}}$) for the CP, EP, and RP for 2-dimensional VASSs. Now the best known lower bounds for these problems are significantly smaller (e.g., NLOGSPACE (NP) for the RP of VASS(2,1,n) (VASS(2,l,n)) [37]). Hence, there is still much room for improvement. However, the two algorithms for the general RP in [24, 29] do not appear to yield better upper bounds for 2-dimensional VASSs. Whether or not these bounds can be tightened we leave as an open question.

Now, before continuing to the detailed discussion, the following definitions are required. Given a VASS=(v_0,V,p_0,S,δ) and a path l in the state graph, $l = s_1 \xrightarrow{v_1} s_2 \xrightarrow{v_2} \cdots s_{t-1} \xrightarrow{v_{t-1}} s_t$ where $s_i \rightarrow (s_{i+1}, v_i)$ $(1 \leq i \leq t-1)$ is in δ, l is a *short loop* iff $s_1 = s_t$ and $s_i \neq s_j$ $(1 \leq i < j \leq t)$. The *displacement* of l, denoted by $|l|$, is $\sum_{i=1}^{t-1} v_i$. l is a *short positive loop* (*p-loop*, for short) iff l is a short loop and $|l| > 0$.

Now the analysis presented in [12] depends heavily on the algorithm given in [11]. So that the reader may comprehend the gist of what follows the algorithm from [11] is listed below. However, only a brief description will be given. The reader is encouraged to refer to [11] for more details. Given a 2-dimensional VASS $\mathcal{V}$, the main idea behind the algorithm is to construct a tree in which each node is labelled by a 3-tuple [x,p,A_x], where $x \in N^2$, $p \in S$ and $A_x \subseteq N^2$, to represent the reachability set generated by $\mathcal{V}$. In what follows, each A_x is called a *loop set*. Each v in A_x is called a *loop vector*. The label $[x,p,A_x]$ indicates that $\{(p,v)\colon v \in \mathcal{L}(x,A_x)\} \subseteq R(v_0,V,p_0,S,\delta)$. The intuitive idea of why the procedure works is the following. The tree is built in such a way that each path, in a sense, corresponds to a path in the VASS. Each time an executable (valid) p-loop is encountered, that particular p-loop will be added (if necessary) to the loop set since clearly that loop can be repeated as many times as we want. If, along any path of the tree, there is an ancestor $[z,p,A_z]$ of $[x,p,A_x]$ such that $A_x=A_z$ and $x\in\mathcal{L}(z,A_z)$, then that particular path terminates at $[x,p,A_x]$. (This condition will be referred to as the *terminating condition*.) In [11], it was shown that a point (p,v) in $S\times N^2$ is reachable in V iff there exists a node with the label $[x,p,A_x]$ such that $v \in \mathcal{L}(x,A_x)$. (In other words, the reachability set coincides exactly with the SLS associated with the tree construction.) Furthermore, the tree construction will eventually terminate. Now, in order to put complexity bounds on this procedure, some measure of the tree is needed. In particular, we will see later that in order to derive the upper bound of the Hopcroft-Pansiot algorithm, it suffices to consider the following two quantities:

(1) $\max\{||v||\colon \exists\ [x,p,A_x]\in T \text{ such that } v\in A_x\}$,

(2) $\max\{||x||\colon [x,p,A_x]\in T\}$.

The first quantity gives an upper bound on the number of periods in each linear set, and the second quantity gives an upper bound on the number of linear sets in the SLS.

<u>Algorithm:</u> (from [11])

```
Create root labelled $[x_0,p_0,\emptyset]$;
while there are unmarked leaves do
begin
    Pick an unmarked leaf $[x,p,A_x]$;
    Add to $A_x$ all displacements of short positive loops from p valid at x;
    if $A_x$ is empty and there exists an ancestor $[z,p,A_z]$ with $z<x$,
    then add x-z to $A_z$;
    if there exists $c \in N^2$, $c=(0,\gamma)$ or $(\gamma,0)$ such that
        (a) c is not colinear to any vector of $A_x$, and
        (b) either
            (i)  there exists an ancestor $[z,p,A_z]$ of $[x,p,A_x]$
                 such that x-z=c, or
            (ii) for some short nonpositive loop from p valid at x
                 with displacement a and some $b \in A_x$,
                 there exists $\alpha, \beta \in N$ such that $\alpha a+\beta b=c$
    then add c to $A_x$;
    if there exists an ancestor $[z,p,A_z]$ of $[x,p,A_x]$ such that
        $\mathcal{L}(z,A_z)$ contains x and $A_z=A_x$
    then mark $[x,p,A_x]$
    else
        for each transition $p \rightarrow (q,v)$ do
        begin
            Let $A_x=\{v_1,\ldots,v_k\}$
            for each a, $a=\alpha_1 v_1+\ldots+\alpha_k v_k$ where
                $(\alpha_1,\ldots,\alpha_k)$ is a minimal k-tuple such that $x+a+v \geq 0$,
            do construct a son $[y,q,A_y]$ where $y=x+a+v$ and $A_x=A_y$;
        end
    if $[x,p,A_x]$ has no son then mark $[x,p,A_x]$;
end
```

Notation: Throughout the remainder of this section, let $\mathcal{V}$ denote an arbitrary VASS with no more than n states or transitions, whose integers can be represented in l bits. Also, let T be its Hopcroft-Pansiot tree.

Now, for some path s in T, one can easily see that there are at most 2^n distinct p-loops in any loop set. In addition to those p-loops, at most one non-axis vector and 2 axis vectors can occur in any of the loop sets in s (in what follows, if they exist, they will be referred to as u_1, γ_1 and γ_2, respectively). Of all the vectors appearing in the loop sets in s, only u_1, γ_1 and γ_2 can have a norm greater than $n*2^l$. Consider an arbitrary path in the tree generated by the algorithm. Let $h_{u_1}(l,n)$, $h_{\gamma_1}(l,n)$ and $h_{\gamma_2}(l,n)$ denote the maximum norm of all the vectors added before u_1, γ_1 and γ_2 are added, respectively. Also, let $h_k(l,n)$ denote the maximum norm of all vectors ever occurring in the system before the k-th loop vector is added. The first step is to derive the quantity $\max\{||v||: \exists\ [x,p,A_x]\in T$ such that $v\in A_x\}$, which is one of the two values we are most interested in. In order to do this, we derive bounds for $||u_1||$, $||\gamma_1||$ and $||\gamma_2||$. In [12], we establish the following facts:

(3) $h_{u_1}(l,n) \leq h_1(l,n)=O(2^{c*l*n})$, and $||u_1||\leq a*2^{b*l*n}$, for some constants a, b, and c independent of l, and n.

(4) $h_{\gamma_1}(l,n)\leq c'*2^{d'*l*n}$, and $||\gamma_1||\leq c*2^{d*l*n}$, for some constants c, c′, d, and d′ independent of l, and n.

(5) $h_{\gamma_2}(l,n)\leq a'*2^{b'*l*n}$, and $||\gamma_2||\leq a*2^{b*l*n}$, for some constants a, a′, b, and b′ independent of l, and n.

Thus, we have a bound for quantity (1). Now from (3), (4), (5), and a straightforward application of the pigeon-hole principle, [12] shows:

(6) $h_k(l,n)=O(2^{k*2^{c*l*n}})$, for some constant c independent of k, l, and n.

Since there will be at most 2^n+3 vectors in any A_x, $\max\{||x||: [x,p,A_x]\in T\}=O(2^{2^{d*l*n}})$, for some constant d independent of $\mathcal{V}$, l, and n. Hence, bounds for both quantities (1) and (2) have been obtained. In addition, [12] gives a VASS in VASS(2,l,n) whose Hopcroft-Pansiot tree can reach a vector with norm $2^{2^{c*l*n}}$, for some positive constant c; hence this upper bound is tight.

As a result of these bounds, in [12] we construct an algorithm to generate a SLS representation of the reachability set of a given VASS. The reader, at this point, should recall that in the original Hopcroft-Pansiot algorithm, no upper bound is given for the size of the SLS representation, nor does it tell how quickly the SLS can be generated. We utilize the bounds on quantities (1) and (2), to construct a modified version of the Hopcroft-Pansiot algorithm. Our algorithm basically expands the tree in the same manner as the Hopcroft-Pansiot algorithm. The modification is that the terminating condition is not checked; instead, no node containing a vector larger than the bound given for quantity (2) is ever added to the tree, and no axis vector larger than the bound derived for quantity (1) is ever added to a loop set. Thus, our tree contains all the nodes in the Hopcroft-Pansiot tree, plus perhaps some additional nodes which are redundant. Furthermore, it is clear that if our tree contains duplicate nodes, then the respective descendants of these nodes are identical. Hence, we can prune our tree by never adding duplicate nodes. Hence, [12] derives:

Theorem 4.1: Given a VASS $\mathcal{V}=(v_0,V,p_0,S,\delta)$ and a state p in S, we can construct a SLS $\mathcal{SL}=\cup_{i=1}^{k}\mathcal{L}_i(x_i,P_i)$ in DTIME($2^{2^{c*l*n}}$), for some constant c independent of $\mathcal{V}$, l, and n, such that,

(a) $\mathcal{SL}=\{x: (p,x)\in R(v_0,V,p_0,S,\delta)\}$,

(b) $k=O(2^{2^{d_1*l*n}})$, for some constant d_1 independent of l and n,

(c) $\forall i$, $1\leq i\leq k$, $||x_i||=O(2^{2^{d_2*l*n}})$, for some constant d_2 independent of l and n,

(d) $\forall i$, $1\leq i\leq k$, $|P_i|=O(2^n)$, where $|P_i|$ is the number of vectors in P_i,

(e) $\forall v\in P_i$, $1\leq i\leq k$, $||v||=O(2^{d_3*l*n})$, for some constant d_3 independent of l and n.

From Theorem 4.1 we want to show that the CP, EP, and RP can be solved in DTIME($2^{2^{c*l*n}}$) for some fixed constant c. While the proof for the RP is quite straightforward, the complexity results for the equivalence problem for SLSs [15, 18] do not directly yield the desired upper bound for the CP and EP. However, a careful application of the proof techniques in [18] yields the desired upper bound for the CP and EP also. See [12]. We therefore have:

Theorem 4.2: For 2-dimensional VASSs, the CP, EP, and RP can be solved in DTIME($2^{2^{c*l*n}}$), for some constant c independent of l and n.

5. The Complexity of Boundedness, Containment, Equivalence, and Reachability for Conflict-Free VRSs

In this section, we describe results from [13, 14] concerning the complexity of the BP, CP, EP, and RP for conflict-free VRSs (VASs, PNs). Conflict-free VASs were introduced by Crespi-Reghizzi and Mandrioli [6]. Landweber and Robertson [25] subsequently showed an upper bound of exponential time for the BP for conflict-free PNs. Since the definitions given in [6, 25] are somewhat different from one another, and since translations between the two do not seem to preserve sharp complexity bounds, we introduced in [14] the notion of conflict-free VRSs (see [23]). Our definition is general enough to include both previous definitions as special cases (although conflict-free PNs have a somewhat more succinct representation), yet it is also restrictive enough to allow us to prove the BP PTIME-complete for all three definitions. In particular, we give in [14] an O($n^{1.5}$) algorithm for deciding boundedness for conflict-free VRSs (or VASs). Due to the fact that a PN is represented somewhat more succinctly, the time complexity is O(n^2) for the PN model--an improvement over the exponential time result of Landweber and Robertson. (For our complexity measures, n is the number of bits needed to encode the problem instance.) Now most problems concerning VRSs or equivalent formalisms have been shown to be intractable (see, e.g., [16, 17, 20, 26, 37]); therefore, since our algorithm has a time complexity of such a small-order polynomial, it may have applications with respect to the verification of parallel computations that can be modelled by conflict-free VRSs.

In [13], we show completeness results concerning conflict-free VRSs for the CP, EP, and RP. Prior to this time, no primitive recursive upper bounds had been given for the complexity of the CP, EP, or RP for conflict-free VRSs. The main result in [13] is that the CP and EP for conflict-free VRSs are Π_2^P-complete, where Π_2^P is the second level of the polynomial-time hierarchy (see Stockmeyer [39]). In showing this, we use a strategy developed by Huynh [18] in showing the equivalence problem for semilinear sets to be in Π_2^P. Given this result and the fact that conflict-free VRSs have semilinear reachability sets [25], one might attempt to solve the problem by translating the VRSs to SLSs and applying Huynh's result directly. However, it can be shown that such a translation must be exponential. Hence, we prove additional properties concerning the resulting SLS representations that enable us to obtain our results via a modified version of Huynh's proof. Now in his proof, Huynh used the fact that the membership problem for semilinear sets is NP-complete [15]. We therefore first show that the RP for conflict-free VRSs is in NP, thus giving the first primitive recursive algorithm for this problem. (Since Jones, Landweber, and Lien [20] have shown the problem to be NP-hard, it follows that the problem is NP-complete.) In order to show this, we give some properties of conflict-free VRSs that allow us to show that there is an instance of integer linear programming that has a solution iff a given instance of the RP has a solution; furthermore, this instance of integer linear programming can be "guessed" in polynomial time. Our next step is to give a SLS representation of the reachability set. We have already mentioned that this representation is exponential in the size of the problem description. On the other hand, we are able to show a high degree of symmetry in the SLS representation. It is this symmetry that allows us to alter Huynh's proof to give our upper bound. Finally, we show a matching lower bound to complete the proof of the main result.

A VRS (v_0,U,V) is said to be *conflict-free* iff (1) no number in U is greater than 1; (2) no number in V is less than -1; (3) no row in V has more than one -1; and (4) if $V(i,j)=-1$, then $U(i,j) = 1$, and all other elements in row i of U are 0. Now there is an obvious translation from a conflict-free PN (see [25]) with k places and m transitions to a $k \times m$ conflict-free VRS whose addition rules and check vectors have no elements larger than 1. Thus, our definition is general enough to include both previous definitions. In addition, all lower bounds shown in this section are shown using VRSs having minimal check matrices and no elements larger than 1. Thus, all of our completeness results hold for conflict-free VRSs, conflict-free VASs, and conflict-free PNs.

5.1. Boundedness

In [14], we present an $O(n^{1.5})$ algorithm to determine boundedness for conflict-free VRSs. Karp and Miller [21] showed that a VRS is unbounded iff it can execute a positive loop. Our algorithm, therefore, looks for a positive loop that can be executed by the VRS. Before a positive loop can be executed, it must be enabled. In [14], we show the following two facts concerning conflict-free VRSs:

(1) For any $k \times m$ conflict-free VRS $\mathcal{V}=(v_0,U,V)$ that is described by n bits, we can construct in time $O(n^{1.5})$ a path σ in which no rule in V is used more than once, such that if some rule v_r is not used in σ, then there is no path in which v_r is used.

(2) Given a conflict-free VRS (v_0,U,V), let V' be any matrix of rules in V such that if $v_j \in V'$, then there is some path that uses v_j. Then there exists a path $\bar{\sigma}$ containing a

loop whose rules used are exactly those in V′ iff every row in V′ that contains a -1 also contains a positive number. Furthermore, the loop in $\bar{\sigma}$ can be required to have each rule in V′ used exactly once.

The path σ described in (1) above is constructed by repeatedly scanning the addition matrix and executing any rule that is enabled and has not yet been executed. It is not hard to show that this procedure operates in $O(n^{1.5})$ time and that the resulting path σ is as described in (1). Furthermore, it is also not hard to show that σ enables the loop described in (2). These two facts, along with results from [6], allow us to produce an $O(n^{1.5})$ algorithm to find a positive loop. We therefore have:

Theorem 5.1: The BP for conflict-free VRSs is solvable in time $O(n^{1.5})$, where n is the number of bits needed to encode the problem instance.

The algorithm given in [14] will also work for conflict-free PNs; however, due to their more succinct representation, we cannot derive the same complexity bound for conflict-free PNs. What we can derive is stated in the following corollary, which gives an improvement over the exponential-time result in [25].

Corollary 5.2: The BP for conflict-free PNs is solvable in $O(n^2)$ time, where n is the number of bits necessary to describe the PN.

Finally, the BP for conflict-free VRSs is shown in [14] to be PTIME-complete. In order to show the problem to be PTIME-hard, [14] uses a reduction from the path system problem, which is known to be PTIME-complete [19]. Hence we have:

Theorem 5.3: The BP for conflict-free VRSs (VASs, PNs, respectively) is PTIME-complete.

5.2. Reachability

The next problem we would like to examine is the RP. Jones, Landweber, and Lien [20] have shown this problem to be NP-hard. Although the problem is known to be decidable [6], no upper bound on its complexity was previously shown. In order to tighten this gap, in [13] we show the problem to be NP-complete. Our strategy is to guess an instance of integer linear programming whose solutions give Parikh maps of sequences of addition rules that lead to the desired vector. The following fact was established in [13]. Together with (1) from Subsection 5.1, it yields sufficient conditions to guarantee that for every solution $\bar{x}$, there is an <u>enabled</u> sequence θ such that $\Psi(\theta)=\bar{x}$.

(1) Let (v_0,U,V) be a $k\times m$ conflict-free VRS, and let θ be an arbitrary sequence of rules from V. If every rule in θ appears in some path that uses only rules from θ, and if $\Delta(\theta)+v_0 \geq \mathbf{0}$, then there is some sequence θ' enabled at v_0 such that $\Psi(\theta')=\Psi(\theta)$.

Using this result, we establish that the RP is NP-complete. Recall that the problem was shown to be NP-hard in [20]. An inspection of the construction used in that proof reveals that it holds for both conflict-free PNs and conflict-free VASs. Our algorithm for deciding reachability works as follows. First, a set of rules to be used in a path is guessed. Let V' be an addition matrix containing exactly these rules. From (1) in Subsection 5.1, it can be verified in polynomial time that exactly these rules can be used in some path. Now from results in Borosh and Treybig [2], we can determine in NP whether $V'x=v-v_0$ has a solution for which each $x(i) \geq 1$. If such a solution exists, it follows from fact (1) above that there is a path from v_0 to v using exactly the rules guessed. We therefore have:

Theorem 5.4: The RP for conflict-free VRSs is NP-complete.

5.3. Containment and Equivalence

We now turn to the CP and EP. In [13], we show that these problems are Π_2^P-complete, where Π_2^P is the set of complements of all languages that can be recognized by a polynomial-time-bounded nondeterministic Turing machine with an NP oracle (see Stockmeyer [39]). In showing the upper bound, we follow a technique used first by Huynh [18] (see also [12]). In [18], Huynh gave a proof that the containment and equivalence problems for semilinear sets are in Π_2^P. Landweber and Robertson [25] have shown that the reachability set of a conflict-free Petri net is semilinear; it is easy to verify that this also holds for VRSs. In [13], we give an upper bound on the size of the SLS representation of the reachability set. In particular, we give an SLS representation in which no integer is larger than $(c*k*m*n)^{d*k*m}$, where k and m are the dimensions of the VRS, n is the largest absolute value of any integer in the VRS, and c and d are fixed constants independent of k, m, and n. Now the technique used in [18] is to show that if the two SLSs are not equal, then there is a "small" witness to that fact. Unfortunately, applying our derived bounds to the result in [18] yields a bound of $O((k*m*n)^{(k*m)^{c'*k^2*m}})$ for the largest integer in the smallest witness. This is clearly too large to guess in polynomial time. Furthermore, we cannot improve our bounds enough to make a direct application of Huynh's results work. To see this, observe that there is a bounded $k\times(k-1)$ conflict-free VRS with start vector (1,0,...,0) which has, for any position i, $2\leq i\leq k$, a rule which will subtract 1 from position i-1 and add 2 to position i. The reachability set of this VRS has at least 2^k bases, and even for SLS representations of this size, Huynh's results yield a bound of $O((k^*m^*n)^{(k^*m)^{c'^*k}})$. In [12], a variation of the proof in [18] was given in which a small enough bound was placed on the sizes of the periods to allow some degree of improvement to be made. However, even if a bound of n could be placed on the largest integer in any period, this proof does not not yield a polynomial bound on the binary representation of the smallest witness. What we are able to do, however, is to give an SLS representation with a high degree of symmetry. It is this symmetry, together with our bound on the size of the SLS representation, that allows us to alter the proof in [18] to give us our tight bound on the size of the smallest witness. The following fact gives the SLS representation of the reachability set of an arbitrary conflict-free VRS.

(1) Let (v_0,U,V) be a $k\times m$ conflict-free VRS in which n is the largest absolute value of any integer. Then there exist constants c_1, c_2, d_1, and d_2, independent of k, m, and n, such that $R(v_0,U,V)=\cup_{v\in B}\mathcal{L}(v,P_v)$, where B is the set of all reachable vectors with no element larger than $(c_1{}^*k^*m^*n)^{c_2{}^*k^*m}$, and P_v is the set of all displacements of loops enabled at v such that if $p\in P_v$, then

   a. p has no element larger than $(d_1{}^*k^*m^*n)^{d_2{}^*k^*m}$, and

   b. if $v(i)=0$, then $p(i)=0$.

We then continue by showing that if some vector w is reachable in a conflict-free VRS $\mathcal{V}_1$ but not in another conflict-free VRS $\mathcal{V}_2$, then there is some vector w' whose binary representation is polynomial in the size of the representations of $\mathcal{V}_1$ and $\mathcal{V}_2$, such that w' is reachable in $\mathcal{V}_1$ but not in $\mathcal{V}_2$. In so doing, we closely follow the technique developed in [18]. An important point to remember is that for any linear set $\mathcal{L}(v,P_v)$ given in the SLS representation of $\mathcal{V}$ (from (1)), $v(i)=0$ only if for all $p\in P_v$, $p(i)=0$. It is precisely this fact that gives us our improvement over a direct application of the results from [18]. Thus, we establish:

(2) There exist constants c and d such that for any two $k\times m$ conflict-free VRSs $\mathcal{V}_1$ and $\mathcal{V}_2$ in which n is the largest absolute value of any integer, if $w\in R(\mathcal{V}_1)\backslash R(\mathcal{V}_2)$, then there exists a $w'\in R(\mathcal{V}_1)\backslash R(\mathcal{V}_2)$ such that $w'(i)\leq(c^*k^*m^*n)^{d^*k^{5*}m}$.

From Theorem 5.4 and (2), we can easily see that the CP and EP are in Π_2^P. Before continuing, we briefly explain our strategy for showing the lower bound. Let X and Y be disjoint sets of Boolean variables, and let F(X,Y) be a Boolean expression in 3DNF. Stockmeyer [39] showed the problem of deciding whether $(\forall X)(\exists Y){:}F(X,Y)=0$ is Π_2^P-complete (the notations $(\forall X)$ and $(\exists Y)$ denote $(\forall x_1 ... \forall x_{n_1})$ and $(\exists y_1 ... \exists y_{n_2})$, respectively, where $X=\{x_1,...,x_{n_1}\}$ and $Y=\{y_1,...,y_{n_2}\}$). In [13], we reduce this problem to the CP and EP. As a result, we obtain:

Theorem 5.5: The CP and EP for conflict-free VRSs are Π_2^P-complete.

Acknowledgment: We would like to thank each of the three referees for their comments, which were helpful when it came to improving the presentation of this paper.

References

[1] Baker, H., Rabin's Proof of the Undecidability of the Reachability Set Inclusion Problem of Vector Addition Systems, MIT Project MAC. CSGM 79, Cambridge, MA, 1973.

[2] Borosh, I. and Treybig, L., Bounds on Positive Integral Solutions of Linear Diophantine Equations, *Proc. AMS 55*, 2 (March 1976), pp. 299-304.

[3] Cardoza, E., Lipton. R. and Meyer, A., Exponential Space Complete Problems for Petri Nets and Commutative Semigroups, *Proceedings of the 8th Annual ACM Symposium on Theory of Computing* (1976), pp. 50-54.

[4] Chan, T., The Boundedness Problem for 3-Dimensional Vector Addition Systems with States, Centre of Computer Studies and Applications, University of Hong Kong, Pokfulam Road, Hong Kong.

[5] Clote, P., The Finite Containment Problem for Petri Nets, *Theoret. Comp. Sci. 43* (1986), 99-106.

[6] Crespi-Reghizzi, S. and Mandrioli, D., A Decidability Theorem for a Class of Vector Addition Systems, *Information Processing Letters 3*, 3 (1975), pp. 78-80.

[7] Davis, M., Matijasevic, Y. and Robinson, J., Hilbert's Tenth Problem. Diophantine Equations: Positive Aspects of a Negative Solution, *Proceedings of Symposium on Pure Mathematics 28* (1976), pp. 323-378.

[8] Ginzburg, A. and Yoeli, M., Vector Addition Systems and Regular Languages, *J. of Computer and System Sciences 20* (1980), pp. 277-284.

[9] Grabowski, J., The Decidability of Persistence for Vector Addition Systems, *Information Processing Letters 11*, 1 (1980), pp. 20-23.

[10] Hack, M., The Equality Problem for Vector Addition Systems is Undecidable, *Theoret. Comp. Sci. 2* (1976), pp. 77-95.

[11] Hopcroft, J. and Pansiot, J., On the Reachability Problem for 5-Dimensional Vector Addition Systems, *Theoret. Comp. Sci. 8* (1979), pp. 135-159.

[12] Howell, R., Rosier, L., Huynh, D., and Yen, H., Some Complexity Bounds for Problems Concerning Finite and 2-Dimensional Vector Addition Systems with States, *Theoret. Comp. Sci. 46* (1986), 107-140.

[13] Howell, R., and Rosier, L., Completeness Results for Conflict-Free Vector Replacement Systems, to be presented at the *14th International Colloquium on Automata, Languages, and Programming,* July, 1987, Karlsruhe, F.R.G. Also Rep. 86-21, The University of Texas at Austin, Austin, Texas, 78712, 1986.

[14] Howell, R., Rosier, L., and Yen, H., An $O(n^{1.5})$ Algorithm to Decide Boundedness for Conflict-Free Vector Replacement Systems, to appear in *Information Processing Letters.*

[15] Huynh, D., The Complexity of Semilinear Sets, *Elektronische Informationsverarbeitung und Kybernetik 18* (1982), pp. 291-338.

[16] Huynh, D., The Complexity of the Equivalence Problem for Commutative Semigroups and Symmetric Vector Addition Systems, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing* (1985), pp. 405-412.

[17] Huynh, D., Complexity of the Word Problem for Commutative Semigroups of Fixed Dimension, *Acta Informatica 22* (1985), pp. 421-432.

[18] Huynh, D., A Simple Proof for the Σ_2^P Upper Bound of the Inequivalence Problem for Semilinear Sets, *Elektronische Informationsverarbeitung und Kybernetik 22* (1986), pp. 147-156.

[19] Jones, N., and Laaser, W., Complete Problems for Deterministic Polynomial Time, *Theoret. Comp. Sci. 3* (1977), pp. 105-117.

[20] Jones, N., Landweber, L. and Lien, Y., Complexity of Some Problems in Petri Nets, *Theoret. Comp. Sci. 4* (1977), pp. 277-299.

[21] Karp, R. and Miller, R., Parallel Program Schemata, *J. of Computer and System Sciences 3*, 2 (1969), pp. 147-195.

[22] Kasai, T., and Iwata, S., Gradually Intractable Problems and Nondeterministic Log-Space Lower Bounds, *Math. Systems Theory 18* (1985), 153-170.

[23] Keller, R.M., *Vector Replacement Systems: A Formalism for Modelling Asynchronous Systems,* TR 117, (Princeton University, CSL, 1972).

[24] Kosaraju, R., Decidability of Reachability in Vector Addition Systems, *Proceedings of the 14th Annual ACM Symposium on Theory of Computing* (1982), pp. 267-280.

[25] Landweber, L. and Robertson, E., Properties of Conflict-Free and Persistent Petri Nets, *JACM 25*, 3 (1978), pp. 352-364.

[26] Lipton, R., The Reachability Problem Requires Exponential Space, Yale University, Dept. of CS., Report No. 62, Jan., 1976.

[27] Mayr, E. and Meyer, A., The Complexity of the Finite Containment Problem for Petri Nets, *JACM 28*, 3 (1981), pp. 561-576.

[28] Mayr, E. and Meyer, A., The Complexity of the Word Problems for Commutative

Semigroups and Polynomial Ideals, *Advances in Mathematics 46* (1982), pp. 305-329.

[29] Mayr, E., An Algorithm for the General Petri Net Reachability Problem, *SIAM J. Comput. 13*, 3 (1984), pp. 441-460. A preliminary version of this paper was presented at the *13th Annual Symposium on Theory of Computing*, 1981.

[30] Mayr, E., Persistence of Vector Replacement Systems is Decidable, *Acta Informatica 15* (1981), pp. 309-318.

[31] McAloon, K., Petri Nets and Large Finite Sets, *Theoret. Comp. Sci. 32* (1984), pp. 173-183.

[32] Minsky, M., *Computation: Finite and Infinite Machines,* (Prentice Hall, Englewood Cliffs, NJ, 1967).

[33] Müller, H., Decidability of Reachability in Persistent Vector Replacement Systems, *Proceedings of the 9th Symposium on Mathematical Foundations of Computer Science*, LNCS 88 (1980), pp. 426-438.

[34] Müller, H., Weak Petri Net Computers for Ackermann Functions, *Elektronische Informationsverarbeitung und Kybernetik 21* (1985), 236-244.

[35] Peterson, J., *Petri Net Theory and the Modeling of Systems,* (Prentice Hall, Englewood Cliffs, NJ, 1981).

[36] Rackoff, C., The Covering and Boundedness Problems for Vector Addition Systems, *Theoret. Comp. Sci. 6* (1978), pp. 223-231.

[37] Rosier, L. and Yen, H., A Multiparameter Analysis of the Boundedness Problem for Vector Addition Systems, *J. of Computer and System Sciences 32*, 1 (February 1986), pp. 105-135.

[38] Savitch, W., Relationships between Nondeterministic and Deterministic Tape Complexities, *J. of Computer and System Sciences 4* (1970), 177-192.

[39] Stockmeyer, L., The Polynomial-Time Hierarchy, *Theoret. Comp. Sci. 3* (1977), 1-22.

[40] Valk, R. and Vidal-Naquet, G., Petri Nets and Regular Languages, *J. of Computer and System Sciences 23* (1981), pp. 299-325.