When Reachability Meets Grzegorczyk

Jérôme Leroux
leroux@labri.fr
Univ. Bordeaux, CNRS, Bordeaux INP, LaBRI, UMR 5800
Talence, France

Abstract

Vector addition systems with states, or equivalently vector addition systems, or Petri nets are a long established model of concurrency with extensive applications in modelling and analysis of hardware, software and database systems, as well as chemical, biological and business processes. The central algorithmic problem is reachability: whether from a given initial configuration there exists a sequence of valid execution steps that reaches a given final configuration. The complexity of the problem has remained unsettled since the 1960s, and it is one of the most prominent open questions in the theory of computation.

In this paper, we survey results about the reachability problem focusing on the general problem. We also show how a recent paper about the reachability problem in fixed dimension combined with vector addition systems with states weakly computing Grzegorczyk hierarchy provides a logspace reduction of the general reachability problem to the bounded case. This result, not included in the original paper due to a lack of space shows that the reachability problem can obviously be decided by a deterministic brute-force exploration. We provide perspectives based on this observation.

CCS Concepts: • **Theory of computation → Logic and verification**.

Keywords: Petri nets, vector addition systems, reachability problem, verification, Grzegorczyk Hierarchy, Ackermannian complexity

ACM Reference Format:
Jérôme Leroux. 2020. When Reachability Meets Grzegorczyk. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS '20), July 8–11, 2020, Saarbrücken, Germany.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3373718.3394732

LICS '20, July 8–11, 2020, Saarbrücken, Germany

ACM ISBN 978-1-4503-7104-9/20/07…$15.00
https://doi.org/10.1145/3373718.3394732

1 Vector Addition Systems with States

Vector addition systems with states [15], or equivalently vector addition systems [17], or Petri nets are one of the most popular formal methods for the representation and the analysis of parallel processes [8]. The central algorithmic problem is reachability: whether from a given initial configuration there exists a sequence of valid execution steps that reaches a given final configuration. Many computational problems reduce to this reachability problem in logic, complexity, real-time systems, protocols [13, 39].

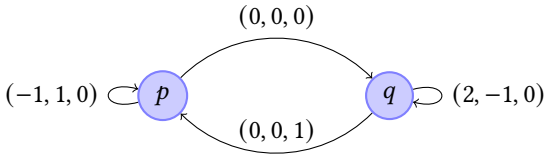
A *vector addition system with states* (VASS for short) of dimension $d \in \mathbb{N}$ is a tuple $V = (Q, q_{ini}, \vec{c}_{ini}, T)$ where Q is a non-empty finite set of states, $q_{ini} \in Q$ is the *initial state*, $\vec{c}_{ini} \in \mathbb{N}^d$ is the *initial vector*, and T is a finite set of *transitions* in $Q \times \mathbb{Z}^d \times Q$. A *configuration* is a pair $(q, \vec{x}) \in Q \times \mathbb{N}^d$ denoted as $q(\vec{x})$ in the sequel. Configuration $q_{ini}(\vec{c}_{ini})$ is called the *initial configuration*. The semantics is defined over *configurations* as follows. With a transition $t \in T$ we associate the binary relation $\xrightarrow{t}$ over the configurations by $p(\vec{x}) \xrightarrow{t} q(\vec{y})$ if $t = (p, \vec{y}-\vec{x}, q)$, where soustraction is performed componentwise. Given a finite word $\pi = t_1 \dots t_k \in T^*$ of transitions, we also define the binary relation $\xrightarrow{\pi}$ over the configurations defined by $p(\vec{x}) \xrightarrow{\pi} q(\vec{y})$ if there exists a sequence $c_0, \dots, c_k$ of configurations such that

$$p(\vec{x}) = c_0 \xrightarrow{t_1} c_1 \cdots \xrightarrow{t_k} c_k = q(\vec{y}) \ .$$

A configuration c is said to be *reachable* if $q_{ini}(\vec{c}_{ini}) \xrightarrow{\pi} c$ for some word $\pi \in T^*$. The set of reachable configurations is called the *reachability set*. We focus in this paper on the reachability problem. This problem consists in deciding the membership of configurations in reachability sets.

Example 1.1. In 1979, Hopcroft and Pansiot [15] introduced the VASS depicted bellow. This VASS has a loop on state p and another loop on state q. Intuitively, iterating the loop on state p transfers the content of the first counter to the second counter whereas iterating the loop on state q transfers and multiplies by two the content of the second counter to the first counter. The third counter is incremented each time we come back to state p from q. In [15] the reachability set from the initial configuration $p(1, 0, 0)$ is proved equal to the following set:

$$\{p(x_1, x_2, x_3) \mid x_1+x_2 \leq 2^{x_3}\} \cup \{q(x_1, x_2, x_3) \mid x_1+2x_2 \leq 2^{x_3+1}\}$$



2 Algorithms for the Reachability Problem

After an incomplete proof by Sacerdote and Tenney [38], decidability of the problem was established by Mayr [33, 34], whose proof was then simplified by Kosaraju [18]. Building on the further refinements made by Lambert in the 1990s [19], there has been substantial progress over the past ten years ultimately proving that the reachability problem can be decided with a simple algorithm based on Presburger inductive invariants [20–22].

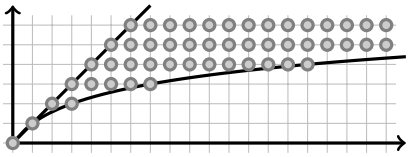
A set C of configurations is called an *inductive invariant* for a VASS if it contains the initial configuration and if $c \xrightarrow{t} c'$ for some $c \in C$ and $t \in T$ implies $c' \in C$. A set C of configurations in $Q \times \mathbb{N}^d$ is said to be *Presburger* if there exists a sequence $(\phi_q)_{q\in Q}$ of formulas ϕ_q in the *Presburger arithmetic* $\mathrm{fo}(\mathbb{N}, +)$ denoting sets $\vec{X}_q \subseteq \mathbb{N}^d$ such that $C = \bigcup_{q\in Q}\{q\} \times \vec{X}_q$.

Theorem 2.1 ([20]). *A configuration c_{out} is not in the reachability set of a VASS V if, and only if, there exists a Presburger inductive invariant for V that does not contain c_{out}.*

Since we can decide if a sequence $(\phi_q)_{q\in Q}$ of Presburger formulas denotes an inductive invariant with classical algorithms deciding the Presburger arithmetic, the previous theorem shows that a brute-force non-deterministic exploration of the reachability set and sequences of Presburger formulas provides a simple algorithm for deciding the reachability problem. Whereas the proof in [20] was based on a refinement of Lambert's algorithm, in [21] a direct proof based on a well quasi order over the executions is provided. This proof was then simplified a bit more in a paper [22] that received a best paper award at Alan Turing centenary conference in 2012. In those two last papers, the Presburer formulas denoting inductive invariants are obtained by proving that reachability sets are *almost semilinear*, a class of sets that extends the class of *semilinear sets* as follows.

A *periodic set* is a subset $\vec{P} \subseteq \mathbb{N}^d$ such that $\vec{0} \in \vec{P}$ and $\vec{P} + \vec{P} \subseteq \vec{P}$. A periodic set $\vec{P}$ is said to be *finitely generated* if there exist vectors $\vec{p}_1, \ldots, \vec{p}_k \in \vec{P}$ such that $\vec{P} = \mathbb{N}\vec{p}_1 + \cdots + \mathbb{N}\vec{p}_k$. A periodic set $\vec{P}$ is said to be *asymptotically definable* if $\mathbb{Q}_{\geq 0}\vec{P}$ is definable in $\mathrm{fo}(\mathbb{Q}_{\geq 0}, +)$. Observe that finitely generated periodic sets are asymptotically definable.

Example 2.2. The periodic set $\vec{P} = \{(p_1, p_2) \in \mathbb{N}^2 \mid p_2 \leq p_1 \leq 2^{p_2} - 1\}$ is depicted on the right. This set is inspired from the previous example 1.1. Observe that $\mathbb{Q}_{\geq 0}\vec{P}$ is the set $\{\vec{0}\} \cup \{(x_1, x_2) \in \mathbb{Q}^2_{>0} \mid x_2 \leq x_1\}$ which is definable in $\mathrm{fo}(\mathbb{Q}_{\geq 0}, +)$.



Let us recall that a set $\vec{X} \subseteq \mathbb{N}^d$ is definable in the Presburger arithmetic if, and only if, it is *semilinear*, i.e. a finite union of *linear sets* $\vec{b} + \vec{P}$ where $\vec{b} \in \mathbb{N}^d$ and $\vec{P} \subseteq \mathbb{N}^d$ is a finitely generated periodic set [11]. The class of *almost semilinear sets* [21] is obtained from the definition of semilinear sets by weakening the finiteness condition on the considered periodic sets. More formally, an *almost semilinear set* is a finite union of sets of the form $\vec{b} + \vec{P}$ where $\vec{b} \in \mathbb{Z}^d$ and $\vec{P} \subseteq \mathbb{Z}^d$ is an asymptotically definable periodic set.

As expected, a set C of configurations is said to be *almost semilinear* if it can be written as $\bigcup_{q\in Q}\{q\} \times \vec{X}_q$ where $\vec{X}_q$ is almost semilinear for every state $q \in Q$. Based on the following geometrical property, in [21, 22] it is proved that Presburger inductive invariants can be obtained by over-approximating almost semilinear sets by semilinear sets.

Theorem 2.3 ([21, 22]). *Intersection of reachability sets with Presburger sets are almost semilinear.*

3 Presburger Reachability Sets

When the reachability set is definable in the Presburger arithmetic, the VASS is said to be Presburger. The existence of sequences of Presburger formulas denoting inductive invariants witnessing the non reachability of a final configuration is trivial in that case since then the reachability set itself is a Presburger inductive invariant. In particular any sequence of Presburger formulas denoting this set is a witness of non reachability for any configuration outside of the reachability set. The problem of deciding if a VASS is Presburger was studied thirty years ago independently by Dirk Hauschildt during his PhD[14] and Jean-Luc Lambert. Unfortunately, these two works were never published. Moreover, from these works, it is difficult to deduce a simple algorithm for computing Presburger formulas denoting the reachability set of Presburger VASSes. In [23] a simple algorithm for computing such a formula based on acceleration techniques is given. Intuitively, acceleration techniques consist in computing the reachability set of a system by computing symbolically the effect of iterating cycles of the system.

More formally, a *path* of a VASS V is a word $\pi \in T^*$ of the form $(q_0, \vec{a}_1, q_1) \ldots (q_{k-1}, \vec{a}_k, q_k)$ where $q_0, \ldots, q_k$ are states in Q and $\vec{a}_1, \ldots, \vec{a}_k$ are vectors in $\mathbb{Z}^d$. When $q_0 = q_k$, the path π is called a *cycle* and it is denoted as θ in the sequel. A *linear path scheme* [30] is a language $L \subseteq T^*$ of paths given by a regular expression of the form $\pi_0\theta_1^*\pi_1 \ldots \theta_k^*\pi_k$

where $\pi_0\theta_1\pi_1\dots\theta_k\pi_k$ is a path for some cycles $\theta_1,\dots,\theta_k$. A *semilinear path scheme* is a finite union of linear path schemes.

A set C included in the reachability set of a VASS V is called *flat* if there exists a semilinear path scheme L such that such that for any configuration $c \in C$ there exists $\pi \in L$ such that:

$$q_{ini}(\vec{c}_{ini}) \xrightarrow{\pi} c$$

A VASS is said to be *flat* if its reachability set is flat.

In [10] heuristics and algorithms for finding (good) semilinear path schemes are provided, and in [31] many subclasses of VASSes with Presburger reachability sets are shown to be flat. In [23], this observation is extended as follows.

Theorem 3.1 ([23]). *Presburger sets included in VASSes reachability sets are flat.*

It follows that the classes of flat VASSes and Presburger VASSes coincide. In particular, the tool FAST [1–3] implemented for analyzing Minsky machines, a class of systems strictly extending VASSes with undecidable reachability problem, is complete and always terminate on the computation of the reachability set of Presburger VASSes.

4 Reversible Reachability Problem

A variant of the VASS reachability problem recently found out many applications in population protocols [7], trace logics [27], universality problems related to structural liveness problems [16], and in solving the home state problem [4], a long standing open problem. The variant, called the *reversible reachability problem*, consists in deciding if two configurations are mutually reachable one from the other.

An instance of the reversible reachability problem can naturally be reduced to two instances of the reachability problem. However, the reversible reachability problem is simplier than the general reachability problem. In fact, in [24] the reversible reachability problem is shown to be exponential-space complete by proving that if two configurations are mutually reachable, then the two configurations belong to a cycle of the (infinite) reachability graph with a length at most doubly-exponential with respect to the size in binary of the two configurations. In [26] this result is refined by focusing on the minimal length of such a cycle, called the distance, with respect to the Euclidean distance between those two configurations. In that paper, it is proved that the distance is linearly bounded by the Euclidean distance upto a doubly-exponential constant that only depends on the VASS. This last result find out a recent application in the computational analysis of an efficient algorithm deciding the VASS coverability problem [9].

5 Complexity of the Reachability Problem

Concerning the complexity of the reachability problem, over the past half century, it has remained unsettled. Lipton's landmark result that the reachability problem requires exponential space [5] has remained the state of the art on lower bounds for over 40 years until it was recently improved to a tower complexity lower-bound in a paper [6] that received a best paper award at the STOC conference in 2019. This result was obtained by introducing the model of *counter programs* that manipulates two kinds of counters: unbounded counters with increment and decrement operations, and counters bounded by a fix parameter k that can be incremented, decremented, but also tested to zero and to the maximal value k. This model is equivalent to the VASSes since the counters that are tested can be hardcoded in the VASS control states.

In that paper, a counter program implementing the following equality is exhibited:

$$n\prod_{i=1}^{k-1}\frac{i+1}{i} = nk$$

Intuitively, the counter program first non-deterministically initializes two unbounded counters x, y to the same positive integer n and then iterates loops that implement weak multiplications of x by the rational numbers $\frac{i+1}{i}$ where i is a bounded counter that ranges over all the values in $\{1,\dots,k-1\}$ (see for instance the cycle on state q in example 1.1 that implements a weak multiplication by 2). By a weak way, we mean that the computed value can be less than or equal to the expected one, but there exists at least one execution that produces the expected maximal value. Thanks to the previous equality, we get a way to check that all the multiplications produce the maximal values. In fact, the counter program implementing the equality finally checks the multiplications with a last loop where counters x and y are respectively decreased simultaneously by k and 1. If all the multiplications produced the maximal values, then just before this last loop, x and y are respectively equal to nk and n. It follows that the last loop succeeds in reaching the zero value on y. Otherwise, the value of x is to small. This trick is the central idea to implement strong multiplication by a rational number. With some additional techniques explained in [6], a tower complexity lowerbound for the reachability problem is obtained.

In 2015, a first complexity upperbound of the reachability problem was provided [28] more than thirty years after the presentation of the algorithm introduced by Mayr [18, 19, 33, 34]. The upperbound given in that paper is cubic Ackermannian. This complexity is obtained by analyzing the computation complexity of the Mayr algorithm. By refining this algorithm and by introducing a new ranking function proving the termination of this refinement, an Ackermannian complexity upperbound was obtained in [29]. This paper also showed that the reachability problem in fixed dimension is primitive recursive by bounding the length of executions thanks to the Grzegorczyk hierarchy.

More formally, the Grzegorczyk hierarchy [12, 32] is defined thanks to a family $(F_d)_{d\in\mathbb{N}}$ of functions $F_d : \mathbb{N} \to \mathbb{N}$ such that every primitive recursive function is asymptotically bounded by some function F_d. This family is defined by $F_0(n) \stackrel{\text{def}}{=} n+1$ and inductively by $F_{d+1}(n) \stackrel{\text{def}}{=} F_d^{n+1}(n)$ for every $n, d \in \mathbb{N}$. Observe that $F_1(n) = 2n+1$, $F_2(n) = 2^{n+1}(n+1)-1$, and $F_3(n)$ grows as a tower of n exponentials. It follows that F_3 is a non elementary function since it eventually exceeds any fixed iteration of the exponential function. An *Ackermannian function*, denoted as F_ω is defined thanks to the diagonal extraction $F_\omega(n) \stackrel{\text{def}}{=} F_{n+1}(n)$ for every $n \in \mathbb{N}$. This function is non primitive recursive.

Theorem 5.1 ([29]). *There exists a constant c such that for every VASS $V = (Q, q_{ini}, \vec{c}_{ini}, T)$ of dimension d, and for every reachable configuration $q_{out}(\vec{c}_{out})$ there exists a word $\pi \in T^*$ such that $q_{ini}(\vec{c}_{ini}) \xrightarrow{\pi} q_{out}(\vec{c}_{out})$ with $|\pi| \leq F_{d+4}(cn)$ where n is the minimal natual number such that:*

- $|Q| \leq n$,
- $\vec{c}_{ini}, \vec{c}_{out} \in \{0, \ldots, n\}^d$, *and*
- $\vec{a} \in \{-n, \ldots, n\}^d$ *for every* $(p, \vec{a}, q) \in T$.

The previous result provides a bound on the minimal length of a word $\pi \in T^*$ to reach a final configuration. In the next section, we show that this bound is sufficient to reduce in logspace the reachability problem to the bounded case. We did not include that result in [29] due to a lack of space.

6 Bounded Reachability Problem

When the reachability set of a VASS is finite, the VASS is said to be *bounded*. The boundedness problem that consists in deciding if a VASS encoded in binary is bounded is known to be exponential space complete since 1978 [5, 37]. The reachability problem for bounded VASSes can be decided by a deterministic brute-force exploration in an obvious way. The computational complexity of such an algorithm is known to be Ackermannian [36]. Moreover, due to the family of VASSes introduced in [35] this bound is optimal. More precisely, in 1981, Mayr and Meyer have exhibited that for each $d \in \mathbb{N}$ we can compute in logspace a VASS of dimension $d+1$ that weakly computes the function F_d.

Based on this family of VASSes and theorem 5.1, let us observe that the reachability problem for VASSes is logspace reducible to the bounded case. In fact, given an instance of the VASS reachability problem, i.e a VASS V of dimension d and a final configuration c_{out}, we can compute in logspace a bounded VASS V' of dimension $2d+5$ and a final configuration c'_{out} such that c_{out} is reachable for V if, and only if, c'_{out} is reachable in V'. Intuitively, VASS V' first weakly computes value $F_d(cn)$ as a budget thanks to the VASS introduced by Mayr and Meyer that weakly computes F_d, and then V' simulates V by decrementing that budget on each simulation step.

It follows that the reachability problem for general VASSes can be solved with a simple deterministic brute-force algorithm. Moreover, it follows that reachability problem for bounded VASSes is a central problem. In fact, the reachability problem for bounded VASSes is equivalent to the reachability problem for general VASSes.

7 Conclusion

We shown in this paper that the reachability problem for bounded VASSes is a central problem since the general reachability problem is logspace inter-reducible to the bounded case. The reachability problem for bounded VASSes is thus a *reachability-complete* problem.

This reachability problem for bounded VASSes can obviously be decided with a deterministic brute-force exploration algorithm. For the time being, this algorithm is the best known algorithm. In fact, the algorithm introduced by Mayr [33, 34], and simplified by Kosaraju [18] and Lambert [19], as well as the new algorithm introduced in [29] fails in improving a brute-force exploration of the reachability set when the VASS is bounded. Intuitively, those algorithms tries to find out cycles that can pump counters to arbitrarily large values. When the reachability set is finite, there is no cycle of that kind. In order to overcome that problem, the notion of *iteration schemes* were recently introduced in [25]. Iteration schemes provide a way to characterize counters that can be pumped to a large, but not arbitrary large value. Finding out a way to use iteration schemes in a reachability algorithm is an open problem.

Finding a better algorithm should be a great breakthrough for understanding the complexity of the reachability problem. The non-deterministic brute-force algorithm based on Presburger inductive invariants seems to be a good candidate. However, the minimal size of formulas denting Presburger inductive invariants is just knwon to be bounded by an Ackermannian function.

Naturally, if the reachability problem is Ackermannian-complete, any attempt to improve the complexity upper-bound of the reachability problem will fail. However those attempt should provide us with insights on how to improve the tower complexity lowerbound. Notice that in that case, the recent algorithm introduced in [29] is optimal, and there is no better algorithm (in the worst case) for solving the reachability problem of bounded VASSes than a brute-force exploration of the reachability set.

Acknowledgments

The author thanks Philippe Schnoebelen and Sylvain Schmitz for remarks about a preliminary version. This work was supported by the grant ANR-17-CE40-0028 of the French National Research Agency ANR (project BRAVAS).

References

[1] S. Bardin, A. Finkel, and J. Leroux. 2004. FASTer Acceleration of Counter Automata in Practice. In *Tools and Algorithms for the Construction and Analysis of Systems, 10th International Conference, TACAS 2004, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2004, Barcelona, Spain, March 29 - April 2, 2004, Proceedings (Lecture Notes in Computer Science)*, Vol. 2988. Springer, 576–590. https://doi.org/10.1007/978-3-540-24730-2_42

[2] S. Bardin, A. Finkel, J. Leroux, and L. Petrucci. 2003. FAST: Fast Acceleration of Symbolikc Transition Systems. In *Computer Aided Verification, 15th International Conference, CAV 2003, Boulder, CO, USA, July 8-12, 2003, Proceedings (Lecture Notes in Computer Science)*, Vol. 2725. Springer, 118–121. https://doi.org/10.1007/978-3-540-45069-6_12

[3] S. Bardin, J. Leroux, and G. Point. 2006. FAST Extended Release. In *Computer Aided Verification, 18th International Conference, CAV 2006, Seattle, WA, USA, August 17-20, 2006, Proceedings (Lecture Notes in Computer Science)*, Vol. 4144. Springer, 63–66. https://doi.org/10.1007/11817963_9

[4] E. Best and J. Esparza. 2016. Existence of home states in Petri nets is decidable. *Inf. Process. Lett.* 116, 6 (2016), 423–427. https://doi.org/10.1016/j.ipl.2016.01.011

[5] E. Cardoza, R. J. Lipton, and A. R. Meyer. 1976. Exponential Space Complete Problems for Petri Nets and Commutative Semigroups: Preliminary Report. In *Proceedings of the 8th Annual ACM Symposium on Theory of Computing, May 3-5, 1976, Hershey, Pennsylvania, USA*. ACM, 50–54. https://doi.org/10.1145/800113.803630

[6] W. Czerwiński, S. Lasota, R. Lazić, J. Leroux, and F. Mazowiecki. 2019. The Reachability Problem for Petri Nets is not Elementary. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*. ACM, 24–33. https://doi.org/10.1145/3313276.3316369

[7] J. Esparza, P. Ganty, J. Leroux, and R. Majumdar. 2017. Verification of population protocols. *Acta Inf.* 54, 2 (2017), 191–215. https://doi.org/10.1007/s00236-016-0272-3

[8] J. Esparza and M. Nielsen. 1994. Decidability Issues for Petri Nets - a Survey. *Bulletin of the European Association for Theoretical Computer Science* 52 (1994), 245–262.

[9] A. Finkel, S. Haddad, and I. Khmelnitsky. 2020. Minimal Coverability Tree Construction Made Complete and Efficient. In *Foundations of Software Science and Computation Structures - 23rd International Conference, FOSSACS 2020, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2020, Dublin, Ireland, April 25-30, 2020, Proceedings (Lecture Notes in Computer Science)*, Vol. 12077. Springer, 237–256. https://doi.org/10.1007/978-3-030-45231-5_13

[10] A. Finkel and J. Leroux. 2002. How to Compose Presburger-Accelerations: Applications to Broadcast Protocols. In *FST TCS 2002: Foundations of Software Technology and Theoretical Computer Science, 22nd Conference Kanpur, India, December 12-14, 2002, Proceedings (Lecture Notes in Computer Science)*, Vol. 2556. Springer, 145–156. https://doi.org/10.1007/3-540-36206-1_14

[11] S. Ginsburg and E. H. Spanier. 1966. Semigroups, Presburger formulas and languages. *Pacific J. Math.* 16, 2 (1966), 285–296. https://doi.org/10.2140/pjm.1966.16.285

[12] A. Grzegorczyk. 1953. *Some classes of recursive functions.* Instytut Matematyczny Polskiej Akademi Nauk. http://eudml.org/doc/219317

[13] M. H. T. Hack. 1975. *Decidability questions for Petri nets.* Ph.D. Dissertation. MIT. http://publications.csail.mit.edu/lcs/pubs/pdf/MIT-LCS-TR-161.pdf

[14] D. Hauschildt. 1990. *Semilinearity of the Reachability Set is Decidable for Petri Nets.* Ph.D. Dissertation. University of Hamburg.

[15] J. E. Hopcroft and J.-J. Pansiot. 1979. On the Reachability Problem for 5-Dimensional Vector Addition Systems. *Theoritical Computer Science* 8 (1979), 135–159.

[16] P. Jancar, J. Leroux, and G. Sutre. 2018. Co-finiteness and Co-emptiness of Reachability Sets in Vector Addition Systems with States. In *Application and Theory of Petri Nets and Concurrency - 39th International Conference, PETRI NETS 2018, Bratislava, Slovakia, June 24-29, 2018, Proceedings (Lecture Notes in Computer Science)*, Vol. 10877. Springer, 184–203. https://doi.org/10.1007/978-3-319-91268-4_10

[17] R. M. Karp and R. E. Miller. 1969. Parallel Program Schemata. *J. Comput. Syst. Sci.* 3, 2 (1969), 147–195. https://doi.org/10.1016/S0022-0000(69)80011-5

[18] S. R. Kosaraju. 1982. Decidability of Reachability in Vector Addition Systems (Preliminary Version). In *STOC*. ACM, 267–281. https://doi.org/10.1145/800070.802201

[19] J.-L. Lambert. 1992. A Structure to Decide Reachability in Petri Nets. *Theor. Comput. Sci.* 99, 1 (1992), 79–104. https://doi.org/10.1016/0304-3975(92)90173-D

[20] J. Leroux. 2010. The General Vector Addition System Reachability Problem by Presburger Inductive Invariants. *Logical Methods in Computer Science* 6, 3 (2010). https://doi.org/10.2168/LMCS-6(3:22)2010

[21] J. Leroux. 2011. Vector addition system reachability problem: a short self-contained proof. In *POPL*. ACM, 307–316. https://doi.org/10.1145/1926385.1926421

[22] J. Leroux. 2012. Vector Addition Systems Reachability Problem (A Simpler Solution). In *Turing-100 (EPiC Series in Computing)*, Vol. 10. EasyChair, 214–228. http://www.easychair.org/publications/paper/106497

[23] J. Leroux. 2013. Presburger Vector Addition Systems. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2013, New Orleans, LA, USA, June 25-28, 2013*. IEEE Computer Society, 23–32. https://doi.org/10.1109/LICS.2013.7

[24] J. Leroux. 2013. Vector Addition System Reversible Reachability Problem. *Logical Methods in Computer Science* 9, 1 (2013). https://doi.org/10.2168/LMCS-9(1:5)2013

[25] J. Leroux. 2018. Polynomial Vector Addition Systems With States. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic (LIPIcs)*, Vol. 107. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 134:1–134:13. https://doi.org/10.4230/LIPIcs.ICALP.2018.134

[26] J. Leroux. 2019. Distance Between Mutually Reachable Petri Net Configurations. In *39th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2019, December 11-13, 2019, Bombay, India (LIPIcs)*, Vol. 150. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 47:1–47:14. https://doi.org/10.4230/LIPIcs.FSTTCS.2019.47

[27] J. Leroux, M. Praveen, and G. Sutre. 2013. A Relational Trace Logic for Vector Addition Systems with Application to Context-Freeness. In *CONCUR 2013 - Concurrency Theory - 24th International Conference, CONCUR 2013, Buenos Aires, Argentina, August 27-30, 2013. Proceedings (Lecture Notes in Computer Science)*, Vol. 8052. Springer, 137–151. https://doi.org/10.1007/978-3-642-40184-8_11

[28] J. Leroux and S. Schmitz. 2015. Demystifying Reachability in Vector Addition Systems. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015*. IEEE Computer Society, 56–67. https://doi.org/10.1109/LICS.2015.16

[29] J. Leroux and S. Schmitz. 2019. Reachability in Vector Addition Systems is Primitive-Recursive in Fixed Dimension. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*. IEEE, 1–13. https://doi.org/10.1109/LICS.2019.8785796

[30] J. Leroux and G. Sutre. 2004. On Flatness for 2-Dimensional Vector Addition Systems with States. In *CONCUR 2004 - Concurrency Theory, 15th International Conference, London, UK, August 31 - September 3, 2004, Proceedings (Lecture Notes in Computer Science)*, Vol. 3170. Springer, 402–416. https://doi.org/10.1007/978-3-540-28644-8_26

[31] J. Leroux and G. Sutre. 2006. Flat counter automata almost everywhere!. In *Software Verification: Infinite-State Model Checking and Static Program Analysis, 19.02. - 24.02.2006 (Dagstuhl Seminar Proceedings)*, Vol. 06081. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany. http://drops.dagstuhl.de/opus/volltexte/2006/729

[32] M. H. Löb and S. S. Wainer. 1970. Hierarchies of number-theoretic functions. I. *Archiv für mathematische Logik und Grundlagenforschung* 13, 1 (1970), 39–51. https://doi.org/10.1007/BF01967649

[33] E. W. Mayr. 1981. An Algorithm for the General Petri Net Reachability Problem. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing, May 11-13, 1981, Milwaukee, Wisconsin, USA*. ACM, 238–246. https://doi.org/10.1145/800076.802477

[34] E. W. Mayr. 1984. An Algorithm for the General Petri Net Reachability Problem. *SIAM J. Comput.* 13, 3 (1984), 441–460. https://doi.org/10.1137/0213029

[35] E. W. Mayr and A. R. Meyer. 1981. The Complexity of the Finite Containment Problem for Petri Nets. *J. ACM* 28, 3 (1981), 561–576. https://doi.org/10.1145/322261.322271

[36] K. McAloon. 1984. Petri Nets and Large Finite Sets. *Theor. Comput. Sci.* 32 (1984), 173–183. https://doi.org/10.1016/0304-3975(84)90029-X

[37] C. Rackoff. 1978. The Covering and Boundedness Problems for Vector Addition Systems. *Theor. Comput. Sci.* 6 (1978), 223–231. https://doi.org/10.1016/0304-3975(78)90036-1

[38] G. S. Sacerdote and R. L. Tenney. 1977. The Decidability of the Reachability Problem for Vector Addition Systems (Preliminary Version). In *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4-6, 1977, Boulder, Colorado, USA*. ACM, 61–76. https://doi.org/10.1145/800105.803396

[39] S. Schmitz. 2016. The complexity of reachability in vector addition systems. *SIGLOG News* 3, 1 (2016), 4–21. https://dl.acm.org/citation.cfm?id=2893585