Abstract

Vector addition systems, or equivalently Petri nets, are one of the most popular formal models for the representation and the analysis of parallel processes. Many problems for vector addition systems are known to be decidable thanks to the theory of well-structured transition systems. Indeed, vector addition systems with configurations equipped with the classical point-wise ordering are well-structured transition systems. Based on this observation, problems like coverability or termination can be proven decidable.

However, the theory of well-structured transition systems does not explain the decidability of the reachability problem. In this presentation, we show that runs of vector addition systems can also be equipped with a well quasi-order. This observation provides a unified understanding of the data structures involved in solving many problems for vector addition systems, including the central reachability problem.

1 Introduction

Vector Addition Systems and Well-Structured Transition Systems. Vector addition systems with states (VASS), or equivalently Petri nets, find a wide range of applications in the modelling of concurrent, chemical, biological, or business processes. They are defined as tuples \( \mathcal{V} = (Q, d, T) \) where \( Q \) is a finite set of states, \( d \) is a dimension in \( \mathbb{N} \), and \( T \) is a finite set of transitions in \( Q \times \mathbb{Z}^d \times Q \) (Figure 1 displays an example). A VASS gives rise to an infinite transition system over the set of configurations \( \text{Confs} \) by allowing a step \( (q, u) \xrightarrow{t} (q', u + a) \) for all \( u \in \mathbb{N}^d \) and \( t = (q, a, q') \in T \) such that \( u + a \geq 0 \). Many problems are decidable for VASS, notably reachability: given \( \mathcal{V} \) and two configurations \( c \) and \( c' \) in \( \text{Confs} \), can \( c \) reach \( c' \) in a finite number of steps, noted \( c \rightarrow^* c' \)?

coverability: given the same inputs, does there exist \( c'' \sqsupseteq c' \) such that \( c \rightarrow^* c'' \)? Here we use the product ordering, i.e. we require \( c' = (q, u') \) and \( c'' = (q, u'') \) where \( u'(i) \geq u''(i) \) for all \( 1 \leq i \leq d \).

These two decision problems form the algorithmic core of many decidability results – spanning from the verification of asynchronous programs [20] to the decidability of data logics [4, 12, 8] (see the references in [48] for more applications).

Vector addition systems are an instance of a more general class of systems with good algorithmic properties called (strict) well-structured transition systems (WSTS), and as
a result several problems are decidable using generic algorithms, including coverability, termination, and boundedness [1, 19]. These algorithms all rely on the existence of a well-quasi-order (wqo) on the set of configurations – here the product ordering \( \sqsubseteq \) over \( \text{Confs} \), which is ‘compatible’ with the transition relation defined by the system at hand.

**Ideals and Complete WSTS.** The theory of WSTS alone does however not account for the decidability of several other problems on VASS, like place boundedness, which asks whether the reachable valuations of an input subset \( K \subseteq \{1, \ldots, d\} \) of the components are bounded. The classical algorithm for this last problem relies instead on the fact that the set of configurations that might be covered starting from some initial configuration \( c \), i.e. the cover (also called the coverability set)

\[
\text{Cover}(c) \overset{\text{def}}{=} \{(q, u) \in \text{Confs} \mid \exists u' \geq u . c \rightarrow^* (q, u')\} = \downarrow\{c' \in \text{Confs} \mid c \rightarrow^* c'\} \tag{1}
\]

is downwards-closed and computable thanks to a coverability tree construction first defined by Karp and Miller [28]. The construction proceeds forwards from \( c \) and explores the tree of reachable configurations, but employs acceleration to ensure finiteness (see Section 3 for details). Due to acceleration, the nodes of this tree are labelled by ‘extended configurations’ in \( Q \times (\mathbb{N} \cup \{\omega\})^d \), where an \( \omega \) value reflects a component that might become arbitrarily large in reachable configurations; the cover is then exactly the union of the downward closures \( \downarrow c \) when \( c \) ranges over the labels in the tree.

The ingredients required to carry out such a construction in general have been identified by Finkel and Goubault-Larrecq [17, 18] with complete WSTS. This framework relies on the existence of

1. an acceleration procedure along finite traces of the system, and of
2. some means to finitely represent downwards-closed sets of configurations. Finkel and Goubault-Larrecq advocate for this the use of ideals, which provide canonical finite decompositions for downwards-closed subsets of a wqo (see Section 2). Finkel and Goubault-Larrecq also provide a range of effective representations for ideals; for instance, the ideals of \((\text{Confs}, \subseteq)\) are exactly the sets \( \downarrow c \) for \( c \in Q \times (\mathbb{N} \cup \{\omega\})^d \) employed in Karp and Miller’s construction.

Based on these two ingredients, the framework of Finkel and Goubault-Larrecq provides a generic procedure to compute a finite representation of the cover – without any general guarantee of termination: the cover is not always computable, e.g. for VASS extended with transfer operations (which are also strict WSTS) the place boundedness problem is undecidable [15].

**The Reachability Problem.** The decidability of the reachability problem for VASS is a famous result, first proven by Mayr [40] in 1981 after years of attempts and partial solutions, notably by Sacerdote and Tenney [46]. Mayr’s algorithm and proof have since been clarified.
J. Leroux and S. Schmitz

and refined by Kosaraju [29] and Lambert [30]; we call the resulting algorithm the KLMST algorithm after its inventors. Put succinctly, this algorithm performs successive refinements on a finite set of structures (called respectively ‘regular constraint graphs’ by Mayr, ‘generalised VASS’ by Kosaraju, and ‘marked graph-transition sequences’ by Lambert), until a condition is fulfilled (called respectively ‘consistent marking’, ‘θ condition’, and ‘perfectness’); at this point the algorithm terminates and can answer whether reachability holds depending on whether the resulting set is empty.

These results have at first sight little to do with WSTS, for which reachability is often undecidable (the case of transfer VASS is again an example). Nevertheless, a recent insight into the algorithm of Mayr, Kosaraju, and Lambert is that they compute an ideal decomposition for the set of runs from source to target configuration. More precisely, we show in [37] that the data structures manipulated in the KLMST algorithm are representations for run ideals, and that the result of the computation is exactly the ideal decomposition of the downward-closure of the set of runs (see Section 4).

Overview of the Talk. To sum up, ideals provide the data structures involved in both

- Karp and Miller’s coverability tree algorithm, which computes the ideal decomposition of the cover using configuration ideals (Section 3), and
- the KLMST algorithm, which computes the ideal decomposition of the downward-closure of the set of runs using run ideals (Section 4).

The purpose of this talk is to present wqo ideals (Section 2) and overview their algorithmic applications through the coverability and KLMST procedures. We believe that the ideal point of view on those two classical algorithms could guide the principled development of algorithms for VASS extensions and other WSTS – in particular when the decidability status of the reachability problem is open, as for unordered data Petri nets [32], branching VASS (e.g. [47]), and pushdown VASS [31, 38]. We shall only provide the basic definitions and main statements here, but we provide pointers to the relevant literature for the interested reader.

2 Ideals for Well-Quasi-Orders

Quasi-Orders. A quasi-order (qo) \((X, \leq_X)\) combines a support set \(X\) with a transitive reflexive relation \(\leq_X \subseteq X \times X\). Given a set \(S\), its downward-closure is \(\downarrow S \overset{\text{def}}{=} \{x \in X \mid \exists s \in S. x \leq_X s\}\); when \(S\) is a singleton \(\{s\}\) we write more simply \(\downarrow s\). A set \(D \subseteq X\) is downwards-closed (also called initial) if \(\downarrow D = D\).

Well-Quasi-Orders. A well-quasi-order (wqo) [23] is a qo with the descending chain property: all the chains \(D_0 \supseteq D_1 \supseteq \cdots\) of downwards-closed subsets \(D_j \subseteq X\) are finite. Equivalently, it has the finite basis property: any subset \(S \subseteq X\) has a finite number of minimal elements. For instance,

finite sets: any finite set \(\Sigma\) equipped with equality forms a wqo \((\Sigma, =)\): its downwards-closed subsets are singletons \(\{x\}\) for \(x \in \Sigma\), and its chains are of length one;

natural numbers: the set of natural numbers \((\mathbb{N}, \leq)\) is a wqo: its downwards-closed subsets are either \(\mathbb{N}\) itself or of the form \(\downarrow n\) for \(n \in \mathbb{N}\), and any chain \((\mathbb{N}, \supseteq)\) \(\downarrow n_0 \supseteq \downarrow n_1 \supseteq \cdots\) corresponds to a decreasing sequence \(n_0 > n_1 > \cdots\) and is therefore finite;

Cartesian products: if \((X, \leq_X)\) and \((Y, \leq_Y)\) are wqos, then their Cartesian product \(X \times Y\) equipped with the product ordering \(\leq_{X \times Y}\) is also a wqo \((X, \leq_X) \times (Y, \leq_Y) \overset{\text{def}}{=} (X \times
Ideal Decompositions for Vector Addition Systems

$Y, \leq_{X \times Y}$, where $(x, y) \leq_{X \times Y} (x', y')$ if and only if $x \leq X x'$ and $y \leq Y y'$ – this allows to prove Dickson’s Lemma [14], which states that $(\mathbb{N}^d, \leq)$ is a wqo when ordered pointwise –;

finite sequences: if $(X, \leq_X)$ is a wqo, then the set $X^*$ of finite sequences over $X$ (sometimes also noted $X^{<\omega}$) equipped with the embedding ordering $\leq_{X^*}$ is also a wqo $(X, \leq_X)^* \overset{def}{=} (X^*, \leq_{X^*})$, where $x_0, \ldots, x_{m-1} \leq_X x'_0, \ldots, x'_{n-1}$ if and only if there exists a monotone injective function $f$ from $\{0, \ldots, m-1\}$ to $\{0, \ldots, n-1\}$ such that $x_j \leq X x'_{f(j)}$ for all $j \in \{0, \ldots, m-1\}$ – this allows to prove Higman’s Lemma [23], which states that $(\Sigma^*, \leq_{\Sigma^*})$ for a finite alphabet $(\Sigma, =)$ is a wqo when ordered by subword embedding.

In the following, we will use these basic examples to construct wqos of VASS configurations (in Section 3) and of VASS runs (in Section 4).

### Ideals

Let $(X, \leq_X)$ be a wqo. An ideal $I$ of $X$ is a non-empty, downwards-closed, and (up-)directed subset of $X$; this last condition enforces that, if $x, x'$ are in $I$, then there exists $y \in I$ that dominates both: $x \leq_X y$ and $x' \leq_X y$.

The key property of wqo ideals we are going to use is that they provide finite decompositions for downwards-closed sets. This was first shown by Bonnet [5], and rediscovered in the context of complete WSTS (and generalised for Noetherian topologies) by Finkel and Goubault-Larrecq [17]:

> **Fact 1 (Canonical Ideal Decompositions).** Every downward-closed set over a wqo is the union of a unique finite family of incomparable (for the inclusion) ideals.

#### Ideal Representations

Combined with the descending chain property, Fact 1 provides an abstract template for algorithms computing descending chains $D_0 \supseteq D_1 \supseteq \cdots$ of downwards-closed sets: this must terminate when working over a wqo, and furthermore each $D_j$ can be represented as a finite set of ideals. The missing element here is how to effectively represent those ideals.

Depending on the wqo at hand, suitable finite representations have been devised in the literature [26, 27, 2]; see [18] for a rather inclusive algebra of such representations. For the basic wqos introduced earlier, this yields:

- **finite sets**: an ideal of $(\Sigma, =)$ is a singleton $\{x\}$ for $x \in \Sigma$; it can be represented by the element $x$ itself with $[x]_{\Sigma} \overset{def}{=} \{x\}$ as associated ideal.

- **natural numbers**: an ideal of $(\mathbb{N}, \leq)$ is either $\mathbb{N}$ itself or a downwards-closed set $\downarrow n$ for $n \in \mathbb{N}$.

They can be represented as elements $x$ of $\mathbb{N} \cup \{\omega\}$ with $[x]_{\mathbb{N}} \overset{def}{=} \downarrow x$ as associated ideal, where we let $\downarrow \omega = \mathbb{N}$.

- **Cartesian products**: an ideal of $X \times Y$ is simply the product of an ideal from $X$ with an ideal from $Y$; hence we can use pairs of representations with $[x, y]_{X \times Y} \overset{def}{=} [x]_X \times [y]_Y$.

- **finite sequences**: an ideal of $X^*$ is a product $P \subseteq X^*$, i.e. a finite concatenation $A_1 \cdot A_2 \cdots A_n$ of atoms $A_j \subseteq X^*$, where the latter are either equal to $\emptyset \cup \{\varepsilon\}$ for some ideal $I$ of $X$ (where $\varepsilon$ denotes the empty sequence), or to $D^*$ for a downwards-closed subset $D$ of $X$ [27].

Products can therefore be represented as simple regular expressions with abstract syntax

$$p ::= a_1 \cdot a_2 \cdots a_n \cdot , \quad a ::= z + \varepsilon \mid (z_1 + \cdots + z_m)^* \quad (2)$$

where $z, z_1, \ldots, z_m$ range over ideal representations for $X$. The associated ideal is defined through the usual semantics for regular expressions:

$$[z + \varepsilon]_{X^*} \overset{def}{=} [z]_X \cup \{\varepsilon\} ,$$

$$[(z_1 + \cdots + z_m)^*]_{X^*} \overset{def}{=} ([z_1]_X \cup \cdots \cup [z_m]_X)^* ,$$

$$[a_1 \cdot a_2 \cdots a_n]_{X^*} \overset{def}{=} [a_1]_{X^*} \cdot [a_2]_{X^*} \cdots [a_n]_{X^*} .$$
Those representations come with algorithms to perform the typically required operations [21], e.g. to check whether \([z]_X \subseteq [z']_X\), or to compute the canonical ideal decomposition of \([z]_X \cap [z']_X\) or \(X \uparrow x\) for any \(x \in X\) and representations \(z, z'\).

### 3 Configuration-Based WQO

Observe that the cover defined in Equation (1) is downwards-closed for \(\sqsubseteq\); it follows that it can be decomposed as a finite union of ideals. In particular, covers can be finitely represented by finite sets of extended configurations, each of them denoting an ideal included in the coverability set. The algorithm of Karp and Miller [28] computes such a representation. We present here in more detail the reasoning leading to this result.

**Ordering Configurations.** The configurations of a VASS are equipped with the product ordering \(\sqsubseteq\):

\[
(\text{Confs}, \sqsubseteq) \overset{\text{def}}{=} (Q, =) \times (\mathbb{N}, \leq)^d. \tag{3}
\]

Rephrased in a more explicit way, \((q, v) \sqsubseteq (q', v')\) if, and only if, \(q = q'\) and \(v(i) \leq v'(i)\) for every \(1 \leq i \leq d\).

**Representing Configuration Ideals.** Notice that \((\text{Confs}, \sqsubseteq)\) is a wqo as a Cartesian product of wqos, and ideals have the following form where \(x \in (\mathbb{N} \cup \{\omega\})^d\):

\[
[(q, x)]_{\text{Confs}} = \{q\} \times \{v \in \mathbb{N}^d \mid v \leq x\}. \tag{4}
\]

Such a pair \((q, x)\) is called an extended configuration and is used as a representation for configuration ideals.

**Extended Steps.** The Karp and Miller algorithm is based on an extension of the step relation \(\rightarrow\) over extended configurations, defined by \((p, x) \xrightarrow{t} (q, y)\) if, and only if, \(t = (p, a, q)\) is a transition in \(T\) for some action \(a\), and for every \(1 \leq i \leq d\):

\[
y(i) = \begin{cases} x(i) + a(i) & \text{if } x(i) \in \mathbb{N}, \\ \omega & \text{otherwise.}\end{cases} \tag{5}
\]

**Coverability Tree Construction.** The Karp and Miller algorithm is computing a tree as follows. Nodes are labelled by extended configurations. Initially, the tree is reduced to a root node labelled by the initial configuration.

A leaf labelled by \(c\) is said to be covered if there exists an ancestor labelled by \(c'\) such that \(c \sqsubseteq c'\). Otherwise the node is said to be uncovered. A leaf labelled by \(c\) is said to be live if \(c \xrightarrow{t} c'\) for some transition \(t\) in \(T\) and some extended configuration \(c'\).

The tree is updated as follows. While there exists a live uncovered leaf, we pick one such leaf \(n\). Assume that \(c = (p, x)\) is the label of \(n\). If there exists an ancestor labelled by \((p, y)\) such that \(y \leq x\) and \(y(i) < x(i) < \omega\) for some \(i\), we pick such an ancestor and we add a child to \(n\) labelled by \((p, z)\) where \(z\) is defined as follows for \(i \in \{1, \ldots, d\}\):

\[
z(i) = \begin{cases} x(i) & \text{if } y(i) = x(i), \\ \omega & \text{if } y(i) < x(i).\end{cases} \tag{6}
\]
This operation, called acceleration, introduces new $\omega$’s on components which intuitively can be increased to arbitrary large values. Otherwise, if there does not exist such an ancestor, for each transition $t$ such that $c \xrightarrow{t} c'$ for some extended configuration $c'$, we add a child to $n$ labelled by $c'$.

**Ideal Decomposition using the Coverability Tree.** The termination of the previous construction relies on the fact that $(\text{Confs}, \sqsubseteq)$ is a wqo. As shown by Karp and Miller [28],

> **Theorem 2.** The Karp and Miller algorithm terminates and it produces a tree satisfying:

$$\text{Cover}(c) = \bigcup_{c' \text{ label of a node}} [c']_{\text{Confs}}.$$  

In particular, by keeping only the maximal labels of the tree, we obtain the unique decomposition of the coverability set into maximal ideals.

> **Corollary 3.** The canonical ideal decomposition of the coverability set is effectively computable.

> **Example 4.** A prefix of the tree computed by the algorithm of Karp and Miller on the 3-dimensional VASS of Figure 1 from the initial configuration $(q_0, 1, 0, 1)$ is depicted in Figure 2. Edges of the tree that introduce $\omega$ using (6) are labelled by ‘$\omega$’. The other ones are labelled by transitions satisfying the extended step relation. Note that the coverability set for this example is quite simple, as it is equal to:

$$[q_0, \omega, \omega, \omega]_{\text{Confs}} \cup [q_1, \omega, \omega, \omega]_{\text{Confs}}.$$  

(7)

In other words, any configuration can be covered in this VASS. This is not so immediate however, as $\omega$s are introduced very progressively in the coverability tree started in Figure 2.

**Applications.** The decomposition of the coverability set into maximal ideals provides a simple algorithm for deciding the coverability problem, since the latter reduces to finding an ideal of the decomposition that contains a given configuration. The decomposition also provides a way to decide many other problems like the place boundedness problem, that takes as input a set $K \subseteq \{1, \ldots, d\}$ and asks whether there exists a bound $m \in \mathbb{N}$ such that every configuration $(q, v)$ reachable from the initial configurations satisfies $v(k) \leq m$ for every $k \in K$. This problem reduces to checking that the extended configurations $(q, x)$ denoting the ideals of the coverability set satisfy $x(k) \in \mathbb{N}$ for every $k \in K$. 

---

![Figure 2](image-url)
Notes on Complexity. From the unique decomposition of the coverability set into maximal ideals, we define the size of the coverability set as the sum of the size of the extended configurations denoting these ideals (with numbers encoded in binary). Since there exists a family of initialised VASS with finite but Ackermannian-sized reachability sets [7], the size of the cover is at least Ackermannian in the worst case. This lower bound is tight, because the Karp and Miller algorithm terminates in at most an Ackermannian number of steps [16].

The algorithm of Karp and Miller is therefore optimal for computing the ideal decomposition of the coverability set. This does not entail that it is optimal for all the problems it can help solving. For instance, on the one hand, the place boundedness problem mentioned earlier can be solved in exponential space [3, 11]. On the other hand, the finite containment problem, which asks given two VASS with finite reachability sets whether the reachability set of the first is included into that of the second, is complete for Ackermannian time [41, 42].

4 Run-Based WQO

Let us denote the set of runs from a source configuration $c$ to a target configuration $c'$ in an input VASS by $\text{Runs}(c, c')$. We denote by $\downarrow \text{Runs}(c, c')$ the downward closure of the set of runs inside a wqo ($\text{PreRuns}$, $\preceq$) defined next in Equation (9), the VASS reachability problem can then be recast as asking whether the downward closed set $\downarrow \text{Runs}(c, c')$ is empty. Since this set is downwards-closed, it can be decomposed into a finite union of ideals, which is computed by the KLMST algorithm. Let us proceed again through the main steps of this result.

Ordering Runs. The set of runs can be partially ordered by introducing the weaker notion of preruns. A prerun is a triple $\rho = (c, w, c')$ where $c$ and $c'$ are two configurations and $w$ is a word over the alphabet $\text{PreSteps} = \text{Confs} \times T \times \text{Confs}$. The configurations $c$ and $c'$ are called respectively the source and target of $\rho$. The set of preruns is denoted by $\text{PreRuns}$. Presteps and preruns are well-quasi-ordered as follows:

$$(\text{PreSteps}, \preceq) \overset{\text{def}}{=} (\text{Confs}, \sqsubseteq) \times (T, =) \times (\text{Confs}, \sqsubseteq)$$

$$(\text{PreRuns}, \preceq) \overset{\text{def}}{=} (\text{Confs}, \sqsubseteq) \times (\text{PreSteps}, \preceq)^* \times (\text{Confs}, \sqsubseteq)$$

A prestep $e = (c, t, c')$ is called a step if it satisfies the step relation $c \xrightarrow{t} c'$. A prerun $(c, w, c')$ is called a run if $w$ satisfies:

- either $w = \varepsilon$ is the empty sequence and then $c = c'$,
- or $w = (c_1, t_1, c'_1) \cdots (c_k, t_k, c'_k)$ is a sequence of steps such that $c = c_1, c' = c'_k$, and $c_{j+1} = c'_j$ for all $1 \leq j < k$.

**Example 5.** Consider again the 3-dimensional VASS of Figure 1. It has a sequence of steps from $c = (q_0, 1, 0, 1)$ to $c' = (q_1, 2, 2, 1)$

$$(q_0, 1, 0, 1) \xrightarrow{t_1} (q_0, 2, 1, 0) \xrightarrow{t_2} (q_0, 1, 1, 1) \xrightarrow{t_1} (q_0, 2, 2, 0) \xrightarrow{t_2} (q_0, 1, 2, 1) \xrightarrow{t_3} (q_1, 2, 2, 1) ,$$

which we see as a run $(c, w, c')$ in $\text{Runs}(c, c')$ with

$$w = ((q_0, 1, 0, 1), (q_0, 2, 1, 0), (q_0, 2, 1, 0), (q_0, 1, 1, 1), (q_1, 2, 2, 0)) \cdots$$

$$(q_0, 2, 2, 0), (q_1, 2, 1, 0), (q_0, 1, 2, 1)) \cdots$$

This is just one example of a run witnessing reachability; observe that any sequence of transitions in

$$\{t_1 t_2 t_3 t_1\}^{n+2} t_3 t_1$$

for $n \geq 0$ would similarly do.
Representing Prerun Ideals. Notice that ideals of \((\text{PreSteps}, \preceq)\) have the following form, where \(c = (c, t, c')\) is an extended prestep, i.e. \(c, c'\) are extended configurations, and \(t \in T\):

\[
\llbracket c \rrbracket_{\text{PreSteps}} = \llbracket c \rrbracket_{\text{Confs}} \times \{t\} \times \llbracket c' \rrbracket_{\text{Confs}} .
\]  

(11)

It follows that ideals of \((\text{PreRuns}, \preceq)\) have the following form, where \(p\) is a regular expression denoting a product over extended steps as defined in Equation (2) and \(c, c'\) are extended configurations:

\[
\llbracket c, p, c' \rrbracket_{\text{PreRuns}} = \llbracket c \rrbracket_{\text{Confs}} \times \llbracket p \rrbracket_{\text{PreSteps}^*} \times \llbracket c' \rrbracket_{\text{Confs}} .
\]  

(12)

Let us instantiate (2) in this case:

\[
p ::= a_1 \cdots a_n , \quad \quad a ::= \varepsilon + e | E^* \]

where \(e\) ranges over extended presteps and \(E\) over finite sets of extended presteps, with semantics \([E^*]_{\text{PreStep}^*} \overset{\text{def}}{=} (\bigcup_{e \in E} \llbracket e \rrbracket_{\text{PreSteps}})^*\). An observation we will use next is that such a set \(E\) can be seen as a finite directed graph with extended configurations \(c\) as vertices, connected by edges labelled by transitions \(t\) in \(T\).

Run Ideals. In [37] we show that the maximal ideals of the decomposition of \(\downarrow \text{Runs}(c, c')\) satisfy some additional properties. More precisely, thanks to the finite basis property of \((\text{PreRuns}, \preceq)\), \(\text{Runs}(c, c')\) has a finite number of minimal elements \(B\) and we can write

\[
\downarrow \text{Runs}(c, c') = \downarrow \left( \bigcup_{\rho \in B} \{\rho' \in \text{Runs}(c, c') \mid \rho \preceq \rho'\} \right) = \bigcup_{\rho \in B} \downarrow \{\rho' \in \text{Runs}(c, c') \mid \rho \preceq \rho'\} .
\]  

(13)

This means we can focus on ideals of the form

\[
\downarrow \{\rho' \in \text{Runs}(c, c') \mid \rho \preceq \rho'\}
\]

(14)

for some run \(\rho\) in \(\text{Runs}(c, c')\). Using the fact that \((\text{Runs}(c, c'), \preceq)\) has the amalgamation property – i.e. if \(\rho \preceq \rho_1\) and \(\rho \preceq \rho_2\) for some runs \(\rho_1, \rho_2 \in \text{Runs}(c, c')\), then there exists a run \(\rho_3 \in \text{Runs}(c, c')\) with \(\rho_1 \preceq \rho_3\) and \(\rho_2 \preceq \rho_3\) – we see that the set in (14) is directed and therefore an ideal.

When considering the representation \((c, p, c')\) for an ideal defined by (14), we then observed in [37] that it has a specific form, which is essentially a syntactic variant of the generalised VASS of Kosaraju [29] and marked graph-transition sequences of Lambert [30]: its product expression \(p\) is of the form

\[
p ::= E_0^* \cdot (e_1 + \varepsilon) \cdot E_1^* \cdots (e_k + \varepsilon) \cdot E_k^* ,
\]  

(15)

i.e. it intersperses extended presteps \(e_j\) and finite sets of extended presteps \(E_j\). Additionally,

- all these extended presteps \(e = (c_1, t, c_2)\) satisfy the extended step relation \(c_1 \xrightarrow{t} c_2\), and
- the graphs defined by the sets \(E_j\) are strongly connected.

Finally, the representation of an ideal like (14) satisfies furthermore a (decidable) adherence condition corresponding to the \(\theta\) condition introduced by Kosaraju [29], or the perfectness condition introduced by Lambert [30]. The point here is that we have now a semantics, in terms of ideals, associated with these syntactic representations and conditions.
Ideal Decomposition using the KLMST Algorithm. Entering the details of the KLMST algorithm would be too long for the purposes of this presentation. We refer to the nice expositions of Müller [43] and Reutenauer [45] for details and examples. The main point here is that the unique decomposition of \( \downarrow \text{Runs}(c, c') \) into maximal ideals is precisely what the KLMST algorithm is computing.

\textbf{Theorem 6 (Decomposition Theorem [37]).} The \textit{KLMST} algorithm computes an ideal decomposition of \( \downarrow \text{Runs}(c, c') \).

Again, by keeping only the maximal ideals in this decomposition, we obtain the unique decomposition of \( \downarrow \text{Runs}(c, c') \) into maximal ideals.

\textbf{Corollary 7.} The canonical ideal decomposition of \( \downarrow \text{Runs}(c, c') \) is effectively computable.

\textbf{Example 8.} Let us come back to the 3-dimensional VASS of Figure 1 and let \( c = (q_0, 1, 0, 1) \) and \( c' = (q_1, 2, 2, 1) \). The ideal decomposition of \( \downarrow \text{Runs}(c, c') \) contains a unique ideal depicted in Figure 3 (see [48] for more details on how this ideal is computed by the KLMST algorithm). This ideal has the following form:

\[
[c, E_0^* \cdot (e_1 + \varepsilon) \cdot E_1^*]_{\text{PresRuns}}
\]  

Figure 3 The unique maximal ideal of \( \downarrow \text{Runs}(c, c') \) for the VASS of Figure 1.

\[ t_2 : (-1, 0, 1) \quad t_1 : (1, 1, -1) \]
\[ q_0, 0, 1, 1 \rightarrow q_0, 1, 0, 1 \rightarrow q_1, 2, 1, 0 \rightarrow q_1, 2, 2, 1 \]

\[ t_3 : (1, 0, 0) \quad t_4 : (0, -1, 0) \]

Applications. Theorem 6 entails the decidability of the reachability problem: \( \text{Runs}(c, c') \) is empty if and only if its downward-closure is. It also allows to prove the completeness of acceleration techniques for computing Presburger definable reachability sets [35].

In the case of labelled VASS where we additionally label transitions in \( T \) by finite sequences over a finite alphabet \( \Sigma \), this also provides a way of constructing the downward-closure of the language between \( c \) and \( c' \), which was already shown to be computable by Habermehl et al. [22] and Zetsche [50].

Notes on Complexity. The decomposition of \( \text{Runs}(c, c') \) into maximal ideals provides a way to associate to this set a size (with numbers encoded in binary). From a complexity point of view, the already mentioned construction of Cardoza et al. [7] shows that, in the worst case, the size of \( \text{Runs}(c, c') \) can be Ackermannian, i.e. is in \( F_\omega(\Omega(n)) \) for an input of size \( n \) (using the fast-growing functions \((F_\alpha)_\alpha\) of Löb and Wainer [39]). We exhibit in
[37] the first upper bound for that size by proving a worst case complexity in $F_{\omega^3}(p(n))$ for an Ackermannian function $p$. This gap between $F_{\omega}(\Omega(n))$ and $F_{\omega^3}(p(n))$ seems difficult to tighten, and the exact complexity is still open. Again, the Ackermannian lower bound on the size of the ideal decomposition does not entail such a gigantic lower bound on the problems it helps solving; in particular, the reachability problem could very well be much simpler, as the best known lower bound is in exponential space [7].

## 5 Conclusion

As we have seen in this short presentation, wqo ideals provide abstract foundations for both the coverability tree construction of Karp and Miller [28] and the KLMST algorithm of Mayr [40], Kosaraju [29], and Lambert [30]. On both accounts, these algorithms compute the canonical ideal decomposition of a downwards-closed set, namely the cover for the coverability tree and the downward-closure of the set of runs for the KLMST algorithm.

This abstract viewpoint on those two algorithms makes them easier to extend to more general classes of systems. In fact, the coverability tree construction has already been extended to unordered data Petri nets [24], branching VASS [49, 25], and pushdown VASS [36]. In each of these cases however, the decidability of the reachability problem is currently open.

Those are not the only algorithmic applications of wqo ideals. For instance, Lazić and Schmitz [33] revisit the usual backward coverability algorithm for WSTS [1, 19] using ideals, and employ it to derive in a uniform manner several known complexity upper bounds on the coverability problem, which were initially based on an approach due to Rackoff [44]: for VASS [6], alternating VASS [9], and branching VASS [13, 34]. Another example is the use of ideal decompositions of formal languages by Zetsche [50], employed for instance by Czerwiński et al. [10] to prove the decidability of separation by piecewise testable languages.

## References


Ideal Decompositions for Vector Addition Systems


