Relational Analysis of Integer Programs[☆,☆☆]

Marius Bozga[a], Radu Iosif[b], Filip Konečný[a,b]

[a] *VERIMAG, CNRS, 2 av. de Vignate, 38610 Gières, France*
[b] *FIT BUT, Božetěchova 2, 61266, Brno, Czech Republic*

Abstract

Verifying an integer program against safety requirements requires, in general, the computation of an invariant of the program, needed to prove the unreachability of one or several error states. Traditionally, such invariants are computed by handling finite representations of potentially infinite sets of states, such as abstract domains, boolean combination of predicates, etc. In this paper, we introduce a method of proving safety properties, that tracks *relations* instead of sets of states. As relations store, in general, more information about the system's behavior than reachability sets, they prove to be a useful tool in designing *modular* verification techniques, in which each function of the program is analysed separately, and its computed summary is plugged in at every call site.

The key to computing accurate relations describing the behavior of a program is inferring the transitive closures of the relations labeling the control loops of the program. We describe an efficient algorithm for computing the transitive closures of *difference bounds*, *octagonal* and *finite monoid affine relations*. On the theoretical side, this framework provides a common solution to the acceleration problem, for all these classes of relations. In practice, according to our experiments, the new method performs up to four orders of magnitude better than existing acceleration algorithms, making it a promising approach for the verification of integer programs. The transitive closure algorithm has been implemented and integrated in a tool for the interprocedural analysis of integer programs.

Keywords: Reachability analysis, Integer programs, Procedure summaries, Transitive closures, Periodic relations

1. Introduction

Integer programs (also known as counter automata, counter systems, or counter machines) are an infinite-state extension of the model of finite-state *boolean programs*,

[☆] This work is an extended version of a conference paper [14] published in Proceedings of CAV'10.
[☆☆] This work was supported by the French national project ANR-09-SEGI-016 VERIDYC, by the Barrande programme (MEB021023), the Czech Science Foundation (projects P103/10/0306 and 102/09/H042), the Czech Ministry of Education (projects COST OC10009 and MSM 0021630528), and the internal FIT BUT grant FIT-S-10-1.

a model which is extensively used in the area of software verification [**?** 36]. The interest for integer programs comes from the fact that they can encode various classes of systems with unbounded (or very large) data domains, such as hardware circuits, cache memories, or software systems with variables of non-primitive types, such as integer arrays, pointers and/or recursive data structures. This comes with no surprise since, in theory, any Turing-complete class of systems can be simulated by integer programs with two variables and increment, decrement and zero test, as shown by Minsky [44]. The reduction used in his proof is however too complex to be used in practice. For practical purposes, a number of recent works have revealed cost-effective reductions of verification problems, for several classes of complex systems, to decision problems, phrased in terms on integer programs. Examples of such systems that can be effectively verified by means of integer programs, include: specifications of hardware components [53], programs with singly-linked lists [10, 26, 13, 16], trees [33], and integer arrays [34, 32, 12]. Hence the growing interest for analysis tools working on integer programs.

Given an integer program, a set of initial states, and a set of error states, the *safety problem* asks whether the program has a computation starting in a state from the initial set, which leads to a configuration from the error set. To answer this question, one typically proceeds by evaluating the set of states reachable from the initial set, and checking the emptiness of the intersection with the set of error states. This method however lacks modularity, which is one of the keys to scalability (i.e., applicability of such algorithms to large systems). Since larger programs are usually organized in many small functions, a modular verification approach aims at running one analysis per function (in isolation), and combining the results in a final verification condition. In order to achieve this goal, one solution is to handle *relations* between states, instead of *sets* of states In this way, each function can be represented by the relation between the input and output valuations of its parameters. The outcome of a function at a certain call site is thus the image of the set of states at that call site, via the function's summary relation. A relational domain seems to be therefore the key ingredient to a modular verification method.

For the purposes of program verification, we consider a domain of relations which are definable in Presburger arithmetic. In general, composing two relations reduces to a quantifier elimination problem, in the underlying theory, and can be solved typically in time polynomial in the sizes of the relations. Computing transitive closures, on the other hand, is a much harder problem, and in general, the formula defining the transitive closure of a linear relation falls outside the known decidable fragments of arithmetic. To this end, it is important to know for which classes of arithmetic relations it is possible to compute the transitive closure precisely and fast. To the best of our knowledge, the three main classes of integer relations for which transitive closures can be computed effectively and precisely are: (1) difference bounds constraints [19, 15], (2) octagons [43, 11], and (3) finite monoid affine transformations [9, 25]. For these three classes, the transitive closures can be moreover defined in Presburger arithmetic[1].

[1]However, these classes are not closed under transitive closures themselves. As we will show, defining the transitive closure of a relation belonging to one of these classes typically requires a combination of linear

The contributions of this paper are two-fold. First, we give a semi-algorithmic method for computing the summary relation of an integer program. The algorithm builds the relation incrementally, by eliminating control states and composing incoming with outgoing relations. The main difficulty here is the elimination of states with several self-loops, for the following reason. Eliminating a state with several self-loops requires computing the transitive closure of a disjunctive relation. Or, if we where able to define the transitive closure of any disjunction of difference bounds relations in a decidable logic (such as Presburger arithmetic), we could solve the reachability problem of any 2-counter machine, a problem which is known to be undecidable [44]. We address this issue by first computing the transitive closures of the self-loops individually, and then exploring all interleavings between them, until no new relations are produced. Obviously, this exploration might not end, in which case we can either use an abstraction (overapproximation) domain to ensure termination, or stop the algorithm at a certain depth, and yield an underapproximation of the exact transitive closure. The first approach can be used to produce correctness certificates, whereas the second is used to find errors in programs.

The main part of this paper is dedicated to presenting a general framework for computing transitive closures of certain relations, called periodic. We define a notion of periodicity on classes of relations that can be naturally represented as matrices. In general, a sequence of integers is said to be *periodic* if the elements of the sequence situated at equal distance one from another differ by the same quantity. This definition is generalized to matrices of integers, entry-wise. A relation R is said to be periodic, if it can be mapped into an integer matrix M_R, such that the sequence $\{M_{R^k}\}_{k=0}^{\infty}$ of the matrix representations of the powers of R, is periodic. For periodic relations, the sequence of powers can be finitely represented, once the period and the rates of the sequence are known. We study the three classes of arithmetic relations mentioned in the previous and show that they are periodic. This provides concise proofs to the fact that the transitive closures for these classes can be effectively computed, and that they are Presburger definable, as initially proved in [19, 11, 9]. Finally, we give EXPTIME upper bounds for the complexity of computing transitive closures of relations belonging to these classes.

Roadmap.

1.1. Related Work

Since the result of Minsky [44], proving Turing-completeness of 2-counter machines with increment, decrement and zero test, research on the verification of integer programs has been pursued in two orthogonal directions. The first one is defining subclasses of systems for which various decision problems are found to be decidable. Examples include Reversal-bounded Counter Machines, [38], Petri Nets and Vector Addition Systems, [51], or Flat Counter Automata, [41]. Often, decidability of various problems is proved by defining the set of reachable configurations in a decidable

arithmetic and modulo constraints.

logic, such as Presburger arithmetic [48]. Such definitions are typically precise, i.e. no information is lost by the use of over-approximation.

A closely related line of work consists in attempts to apply Model Checking [18, 50] techniques to the verification of infinite-state systems. Such techniques consider the problem of accelerating transition relations by successive under-approximations. The methods based on acceleration are not guaranteed to terminate, in general. However, for certain restricted classes of systems, one can prove termination of such verification methods. These classes are typically equivalent, from a semantical point of view, with *flat* systems, in which the control loops are executed in a certain partial order [6].

For systems with integer variables, the acceleration of affine relations has been considered primarily in the works of Annichini et. al [4], Boigelot [9], and Finkel and Leroux [25]. Finite monoid affine relations have been first studied by Weber and Seidl [**?**] and Boigelot [9], who shows that the finite monoid property is decidable, and that the transitive closure is definable in Presburger arithmetic, in this case. On what concerns non-deterministic transition relations, difference bounds constraints appear in the context of the verification of systems modeled using timed automata [1].

The transitive closure of a difference bounds constraint is shown to be Presburger definable first by Comon and Jurski [19]. Their proof was subsequently simplified and extended to parametric difference bounds constraints in our previous work [15]. Subsequently, we showed that octagonal relations can be accelerated, and that the transitive closure is also Presburger definable [11] in this case. The proofs of ultimate periodicity from this paper are based on some of our previous results [15, 11]. For difference bounds constraints, the proof from [15] was simplified using a result from tropical semiring theory [52].

Another, orthogonal, direction of work is concerned with finding sound (but not necessarily complete) answers to the decision problems mentioned above, in a cost-effective way. Such approaches, based on the theory of Abstract Interpretation [21], use abstract domains (such as e.g., polyhedra [22], octagons [43], etc.) and compute fixed points of the transfer functions, which are overapproximations of the sets of reachable configurations. The drawback of the methods based solely on Abstract Interpretation is the inability to deal with false positives i.e., errors caused by the use of a too coarse abstract domain. Typically, in these cases, ruling out spurious counterexamples requires a fair amount of human experience.

The method of Predicate Abstraction [31] combines ideas from Abstract Interpretation and Model Checking in order to compute program invariants in a goal-driven fashion, namely by applying refinement techniques (such as Craig Interpolation [**?**]) to rule out spurious counterexamples. This technique is also known as Counter Example-based Abstraction Refinement (CEGAR) [17]. On one hand, cutting edge CEGAR tools, such as e.g. ARMC [47], BLAST [36] or CPA [8], appear to be quite effective in finding bugs and certifying correctness of real-life systems (device drivers, web servers, operating system kernels). On the other hand, goal-driven search is not easily amenable to cope with modular verification, since the reason for spuriously reaching an error state might reside in the over-approximation of the behavior of a function call. Since the error location is typically not part of that function, it is usually hard to trace the relation between the cause and effect, in order to refine the abstraction in the right way. A method that attempts to apply predicate abstraction to programs composed of (possi-

bly recursive) functions is the method of *nested interpolants* [35]. This method lacks however modularity, as it represents the entire programs by a nested word automata [3] i.e., computation models which are equivalent to the visible pushdown automata [2].

Our work focuses on modular program verification, by attempting to compute function summaries, without regard to the calling context. On one side, unlike the techniques based on Abstract Intepretation, we aim at computing precise summary relations, that should not require refinement. On the other side, our method, although modular, is computationally more expensive than the error-driven search of predicate abstraction, mostly due to the lack of abstraction (and refinement) in our method. A future combination of these two (apparently antithetical) approaches to program verification seems to be the key to a wider application of program verification in real-life software development. The idea of using relations as a domain of program analysis has been also exploited in [46], although with the goal of proving program termination, rather than safety, which is the purpose of the present paper.

2. Motivating Example

We start with an example of an integer program for which we prove safety (unreachability of the error state) by computing the relation between the values of its variables at the initial and final control locations. For simplicity reasons, we consider that a program is represented by a *control flow graph*, whose vertices are control locations, and whose edges are labeled with conjunctions of (in)equalities between linear terms. Intuitivelly, given an edge of the control flow graph, which is labeled with a statement, an unprimed variable name represents the value of the variable at the source location, while a primed variable name tracks the value of the variable at the destination location. Consider, for instance, the program in Figure 1a and the control flow graph of the `fun` function, in Figure 1b and 1c.

The function defined in Figure 1a takes two integer parameters and returns an integer result. The goal of the analysis is to prove that the assertion at line 9 holds, whenever the control reaches it. A non-modular method will typically iterate the loop at lines 8 and 9 and analyse the behavior of the `fun` function for each different valuation of the formal parameters `x` and `m`. Our method instead will first infer the transfer relation for the function, and then will use this relation in proving all assertions correct at once.

This method is similar to the classical conversion of finite automata into regular expressions. We proceed by eliminating the control states from the control flow graph of the `fun` function (Figure 1b), and recording the result of these eliminations on the remaining edges. We start by first eliminating the states without self-loops. Each elimination requires composing all incoming with all outgoing edge relations – the edges labeled by inconsistent relations are not added. First, we eliminate l_3 and l_4, in Figure 2a, followed by l_5 and l_6, in Figure 2b. Next, we eliminate l_2, which causes l_1 to have two self-loops, labeled by the relations $R_1 \Leftrightarrow x < n \wedge x < m \wedge x' = x+1 \wedge y' = y+1$ (the left loop in Figure 2c) and $R_2 \Leftrightarrow x < n \wedge x \geq m \wedge x' = x+1 \wedge y' = y-1$ (the right loop in Figure 2c).

At this point, we need to eliminate a control location (l_1) with two self-loops. This step requires the computation of the transitive closure of the disjunctive relation corre-

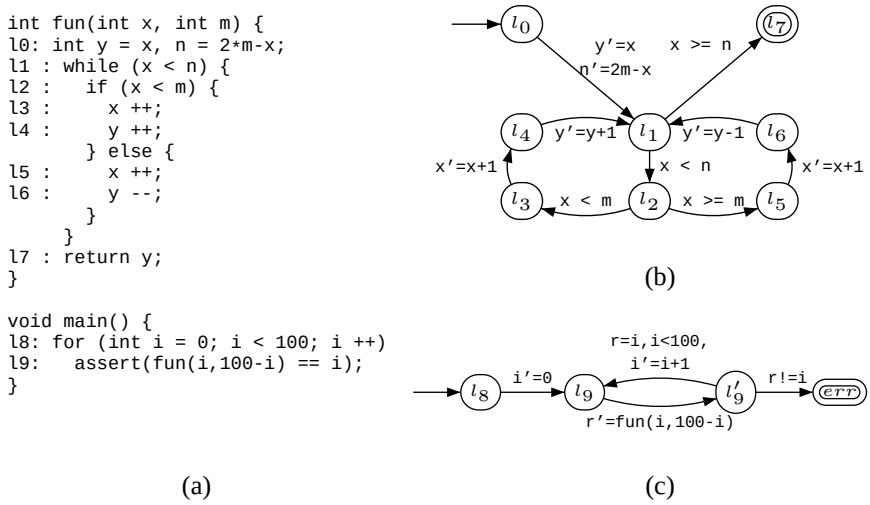


Figure 1: Example of an integer program and its control flow graph

sponding to iterating these loops in any possible order. In general, this computation is not bound to yield a result that can be expressed in linear arithmetic (or, for that matter, in any decidable subfragment of first-order arithmetic). Hence, our method consists of a semi-algorithm.

We first compute transitive closures of the conjunctive relations R_1^+ and R_2^+ (Figure 2e). Then we systematically explore all possible interleavings of R_1^+ and R_2^+, by building a tree (breadth-first) whose edges are labeled with the transitive closures of R_1^+ and R_2^+, and each node corresponds to the composition of the transitive closures on the path from the root (labeled with the identity relation $\mathcal{I}$) to the node. Before expanding the tree, the algorithm checks whether the new relation is either (i) inconsistent or (ii) included in the union of the relations labeling the existing nodes. If this test succeeds, the algorithm backtracks, otherwise it adds the new node to the tree. If the transitive closure $(R_1 \vee R_2)^+$ is equivalent to a finite number of interleavings, this construction will terminate, otherwise not. For this example, in Figure 2d, the tree construction ends after three iterations, with the result $(R_1 \vee R_2)^+ = R_1^+ \vee R_2^+ \vee R_1^+ \circ R_2^+$, see Figure 2e. This is because the composition $R_2^+ \circ R_1^+$ is inconsistent. Next, the location l_1 is split into l_1' and l_1'' (this time, both locations have no self-loops), and there are four edges between l_1' and l_1'', labelled with $\mathcal{I}$, R_1^+, R_2^+ and $R_1^+ \circ R_2^+$, see Figure 2f. We further eliminate the locations l_1' and l_1'', see Figure 2g. Finally we eliminate variables not appearing in the signature of the function and obtain the transfer relation of the `fun` function: $(x \geq m \wedge y' = x) \vee (x < m \wedge y' = x) \equiv (y' = x)$.

This relation is now used to check the validity of the assertion at line 9, in Figure 1a. Since the call to `fun` on line 9 can be replaced by the summary relation computed above, we can represent the `main` function by a control flow graph, and apply the state elimination method in order to establish that the error state (corresponding to a

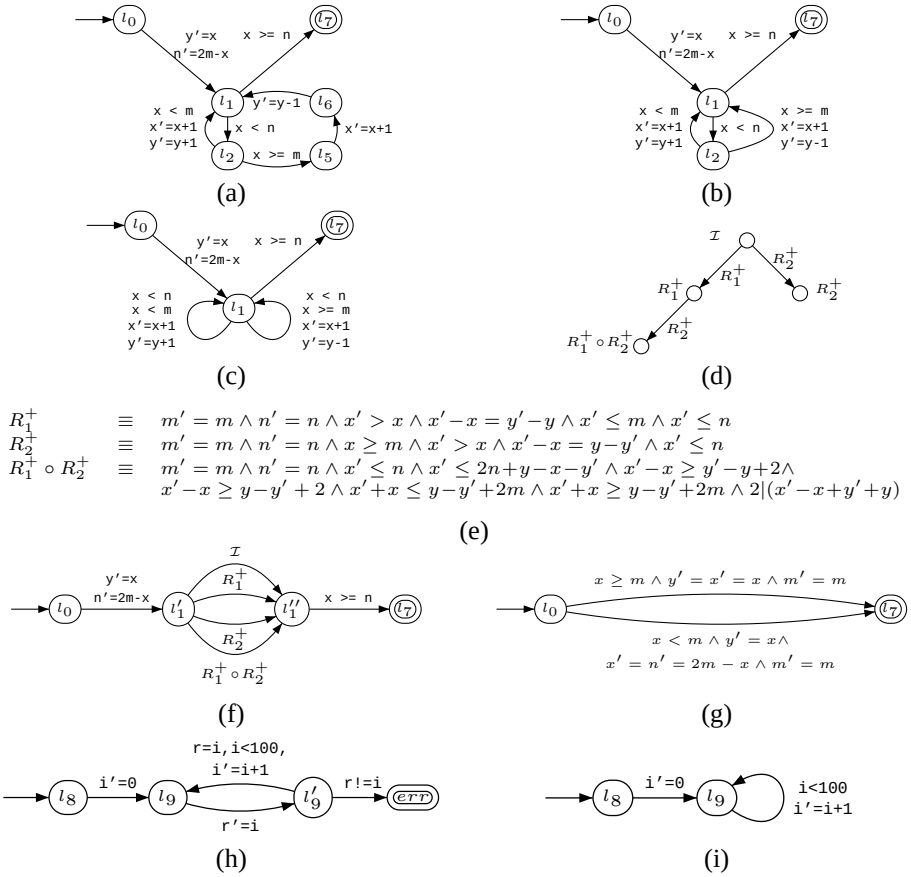


Figure 2: Deciding Safety by Elimination of Control Locations

violation of the assertion) is unreachable. First, we substitute the the call arguments and return values to the transfer function, see Figure 2h. Then we eliminate control state l'_9. Since the composition of the transitions $l_9 \xrightarrow{r'=i} l'_9$ and $l'_9 \xrightarrow{r\neq i} err$ is unsatisfiable, the error state is unreachable (Figure 2i). We can now conclude that the program is safe with respect to the assertion at line 9.

3. Preliminary Definitions

We denote by $\mathbb{Z}$, $\mathbb{N}$ and $\mathbb{N}_+$ the sets of integers, positive (including zero) and strictly positive integers, respectively. We denote by $\mathbb{Z}_\infty$ and $\mathbb{Z}_{-\infty}$ the sets $\mathbb{Z} \cup \{\infty\}$ and $\mathbb{Z} \cup \{-\infty\}$, respectivelly. In the rest of this paper we will fix the set of variables $\mathbf{x} = \{x_1, x_2, \ldots, x_N\}$, for some $N > 0$. The set of *primed* variables is $\mathbf{x}' = \{x'_1, x'_2, \ldots, x'_N\}$. These variables are assumed to be ranging over $\mathbb{Z}$, unless otherwise

specified. For a set $S \in \mathbb{Z}$ of integers, we denote by $\max S$ the largest integer $m \in S$, if one exists. By $\sup S$ we denote the smallest value $m \in \mathbb{Z}_\infty$ such that $s \leq m$, for all $s \in S$.

A *linear term* t over a set of variables in $\mathbf{x}$ is a linear combination of the form $a_0 + \sum_{i=1}^{n} a_i x_i$, where $a_0, a_1, \ldots, a_n \in \mathbb{Z}$. An *atomic proposition* is a predicate of the form either $t \leq 0$, or $t \equiv_c 0$, where t is a linear term, $c \in \mathbb{N}_+$ is a strictly positive integer, and $\equiv_c$ is the equivalence relation modulo c. *Presburger arithmetic* is the first-order logic over propositions $t \leq 0$; Presburger arithmetic has quantifier elimination and is decidable [48]. For simplicity we consider only formulas in Presburger arithmetic in this paper.

For a first-order logical formula φ, let $FV(\varphi)$ denote the set of its free variables. By writing $\varphi(\mathbf{x})$ we imply that $FV(\varphi) \subseteq \mathbf{x}$. For a formula $\varphi(\mathbf{x})$, we denote by $\varphi[t_1/x_1, \ldots, t_N/x_N]$ the formula obtained from φ by syntactically replacing each free occurrence of $x_1, \ldots, x_N$ with the terms $t_1, \ldots, t_N$, respectively. We denote by $\overrightarrow{\mathbf{y}} = \langle y_1, \ldots, y_k \rangle$ an ordered sequence of variables. We denote by $|\overrightarrow{\mathbf{y}}| = k$ the length of $\overrightarrow{\mathbf{y}}$. By $\overrightarrow{\mathbf{y}} = \overrightarrow{\mathbf{z}}$, where $|\overrightarrow{\mathbf{y}}| = |\overrightarrow{\mathbf{z}}| = k$, we denote the formula $\bigwedge_{i=1}^{k} y_k = z_k$. If $\mathbf{x}$ is a set of variables, we write $\overrightarrow{\mathbf{y}} \subseteq \mathbf{x}$ if all elements of $\overrightarrow{\mathbf{y}}$ are in $\mathbf{x}$.

A *valuation* of $\mathbf{x}$ is a function $\nu : \mathbf{x} \to \mathbb{Z}$. The set of all such valuations is denoted by $\mathbb{Z}^{\mathbf{x}}$. If $\overrightarrow{\mathbf{y}} = \langle y_1, \ldots, y_k \rangle$ is an ordered sequence of variables, we denote by $\nu(\overrightarrow{\mathbf{y}})$ the sequence of integers $\langle \nu(y_1), \ldots, \nu(y_k) \rangle$. If $\nu \in \mathbb{Z}^{\mathbf{x}}$, we denote by $\nu \models \varphi$ the fact that the formula obtained from φ by replacing each occurrence of x_i with $\nu(x_i)$ is valid. Similarly, an arithmetic formula $R(\mathbf{x}, \mathbf{x}')$ defining a relation $R \subseteq \mathbb{Z}^N \times \mathbb{Z}^N$ is evaluated with respect to two valuations ν_1 and ν_2. The satisfaction relation is denoted $(\nu_1, \nu_2) \models R$. By $\models \varphi$ we denote the fact that φ is *valid* i.e., logically equivalent to true. We say that an arithmetic formula $\varphi(\mathbf{x})$ is *consistent* if there exists a valuation ν such that $\nu \models \varphi$. We use the symbols $\Rightarrow, \Leftrightarrow$ to denote logical implication and equivalence, respectively. The consistency of a formula φ is usually denoted by writing $\varphi \not\Leftrightarrow$ false.

The composition of two relations $R_1, R_2 \in \mathbb{Z}^N \times \mathbb{Z}^N$ is denoted by $R_1 \circ R_2 = \{(\mathbf{u}, \mathbf{v}) \in \mathbb{Z}^N \times \mathbb{Z}^N \mid \exists \mathbf{t} \in \mathbb{Z}^N \,.\, (\mathbf{u}, \mathbf{t}) \in R_1 \text{ and } (\mathbf{t}, \mathbf{v}) \in R_2\}$. Let ϵ_N be the identity relation $\{(\mathbf{u}, \mathbf{u}) \mid \mathbf{u} \in \mathbb{Z}^N\}$. For any relation $R \in \mathbb{Z}^N \times \mathbb{Z}^N$, we define $R^0 = \epsilon_N$ and $R^i = R^{i-1} \circ R$, for any $i > 0$. With these notations, $R^+ = \bigcup_{i=1}^{\infty} R^i$ denotes the *transitive closure* of R, and $R^* = R^+ \cup \epsilon$ denotes the *reflexive and transitive closure* of R. The pre- and post-images of a set $S \subseteq \mathbb{Z}^N$ via a relation $R \subseteq \mathbb{Z}^N \times \mathbb{Z}^N$ are defined as $\mathrm{pre}(S, R) = \{\mathbf{u} \in \mathbb{Z}^N \mid \exists \mathbf{v} \in S \,.\, (\mathbf{u}, \mathbf{v}) \in R\}$ and $\mathrm{post}(S, R) = \{\mathbf{v} \in \mathbb{Z}^N \mid \exists \mathbf{u} \in S \,.\, (\mathbf{u}, \mathbf{v}) \in R\}$. The *weakest pre-image* is the dual of the pre-image $\widetilde{pre}(S, R) = \{\mathbf{u} \in \mathbb{Z}^N \mid \forall \mathbf{v} \,.\, (\mathbf{u}, \mathbf{v}) \in R \Rightarrow \mathbf{v} \in S\}$. We sometimes denote relations by their defining logical formulae.

An *idempotent semiring* is a set $(\mathcal{S}, +, \cdot, \mathbf{0}, \mathbf{1})$ equipped with two operations, the addition $+$ and the multiplication $\cdot$, such that $(\mathcal{S}, +, \mathbf{0})$ is an idempotent (i.e., $p + p = p$ for all $p \in \mathcal{S}$) commutative monoid with neutral element $\mathbf{0}$ and $(\mathcal{S}, \cdot, \mathbf{1})$ is a monoid with neutral element $\mathbf{1}$. Moreover, multiplication distributes both left and right over addition and $\mathbf{0} \cdot r = r \cdot \mathbf{0} = \mathbf{0}$, for all $r \in \mathcal{S}$.

4. Modular Safety Analysis of Integer Programs

In this section we describe a modular verification technique for integer programs consisting of collections of procedures with integer parameters, that call each other. Our method is able to analyze a function in isolation, independently of its calling context, and infer a summary relation, between the input and output values of its parameters. On the downside, recursive call schemes are currently beyond the reach of our technique, being considered as subject of future research.

In the rest of this section, we simplify the form of integer programs, without loss of generality, in several ways. First, we consider that the semantics of all statements of the program are relations that can be encoded in Presburger arithmetic. The idea is that a program statement which applies multiplication to two different variables can be simulated by a program using only Presburger definable statements[2]. Second, we assume that each statement can be encoded using a quantifier-free conjunction of atomic propositions – disjunctive relations can be represented using different transitions between the same source and destination control location. Finally, we assume that the transitive closure of each conjunctive relation corresponding to a cycle in the program is Presburger definable. As we show in the next section, this assumption holds for certain classes of relations, called *periodic*. Since each statement on a loop can be simulated using only increment, decrement and zero test, and moreover, these relations are all periodic, as it will be shown next, the assumption on the computability of transitive closures of cycles loses no generality.

4.1. Syntax

We define integer programs as collections of procedures. We abstract from specific programming language constructs and assume that each procedure is a control flow graph whose edges are labeled by Presburger arithmetic relations. In addition, certain edges correspond to calls between procedures, and the parameters and results are passed on by values. Formally, an *integer program* is a tuple $\mathcal{P} = \langle \mathbf{x}_g, \{P_1, \ldots, P_n\}, P_m \rangle$ where $\mathbf{x}_g$ are *global* variables, $P_1, \ldots, P_n$ are *procedures*, P_m is the *main* procedure, for some $m = 1, \ldots, n$, and each procedure is a tuple $P_i = \langle \mathbf{x}_i, \overrightarrow{\mathbf{x}}_i^{in}, \overrightarrow{\mathbf{x}}_i^{out}, Q_i, q_{0,i}, q_{f,i}, q_{e,i}, \Delta_i \rangle$, where:

- $\mathbf{x}_i$ are the *local* variables of P_i. We require that $\mathbf{x}_i \cap \mathbf{x}_g = \emptyset$ and that $\mathbf{x}_i \cap \mathbf{x}_j = \emptyset$ for all indices $i \neq j$, i.e. the global variables are the only variables shared by any two procedures.
- $\overrightarrow{\mathbf{x}}_i^{in} \subseteq \mathbf{x}_i$ and $\overrightarrow{\mathbf{x}}_i^{out} \subseteq \mathbf{x}_i$ are the *input* and *output* variables of P_i. Intuitively, input variables are used to pass the arguments, and output variables are used to retrieve the resulting values from a procedure.
- Q_i are the *control states* of P_i. We require that the sets of control states are pairwise disjoint, i.e. $Q_i \cap Q_j = \emptyset$, for all $i \neq j$.

[2] In fact, only increment, decrement and zero test are sufficient.

- Δ_i is a set of *transition rules* of the form, either:
  1. $q \xrightarrow{R(\mathbf{x}_i \cup \mathbf{x}_g, \mathbf{x}'_i \cup \mathbf{x}'_g)} q'$ is an *internal transition*, where $q, q' \in Q_i$ are the source and destination state, and $R(\mathbf{x}_i \cup \mathbf{x}_g, \mathbf{x}'_i \cup \mathbf{x}'_g)$ is a Presburger arithmetic relation
  2. $q \xrightarrow{\overrightarrow{\mathbf{z}'} = call_j(\overrightarrow{\mathbf{t}})} q'$ is a *call transition*, where $q, q' \in Q_i$ are the source and destination control states, respectively, $j = 1 \dots n$ is the index of the callee procedure, $\overrightarrow{\mathbf{t}}$ is a sequence of linear terms over $\mathbf{x}_g \cup \mathbf{x}_i$, called *parameters*, and $\overrightarrow{\mathbf{z}} \subseteq \mathbf{x}_g \cup \mathbf{x}_i$ is a sequence of variables, called *return variables*. We require that $|\overrightarrow{\mathbf{t}}| = |\overrightarrow{\mathbf{x}}_j^{in}|$ and $|\overrightarrow{\mathbf{z}}| = |\overrightarrow{\mathbf{x}}_j^{out}|$, i.e. the numbers of parameters and return variables of the call transition match the numbers of input and output variables of the callee, respectively.

- $q_{0,i}, q_{f,i}, q_{e,i} \in Q_i$ are the *initial*, *final* and *error* control states of P_i. We require that these states are pairwise disjoint, that $q_{0,i}$ has no incoming transition rules, and that $q_{f,i}$ and $q_{e,i}$ have no outgoing transition rules.

For a program $\mathcal{P} = \langle P_1, \dots, P_n \rangle$ the *call graph* of $\mathcal{P}$, denoted $CG(\mathcal{P}) = \langle \mathcal{P}, \hookrightarrow \rangle$, is a graph whose vertices are procedures, and edges $P_i \hookrightarrow P_j$ represent calls of P_i to P_j. The program $\mathcal{P}$ is said to be *recursive* if and only if $CG(\mathcal{P})$ has cycles. In this thesis, we proceed under the assumption that the considered program is not recursive.

4.2. Semantics

A *configuration* of a procedure $P_i = \langle \mathbf{x}_i, \overrightarrow{\mathbf{x}}_i^{in}, \overrightarrow{\mathbf{x}}_i^{out}, Q_i, q_{0,i}, q_{f,i}, q_{e,i}, \Delta_i \rangle$ is a pair $\langle q, \nu \rangle$, where $q \in Q_i$ is a control state and $\nu : \mathbf{x}_i \cup \mathbf{x}_g \to \mathbb{Z}$ is a valuation of the variables visible in P_i. For each procedure P_i, we define the set of valuations of variables visible in P_i as $\mathcal{V}_i = \mathbb{Z}^{\mathbf{x}_i \cup \mathbf{x}_g}$. Next, we define predicates MATCHCALL and MATCHCALLRET which are later used to define compatibility of valuations at call (return) sites with initial (final) valuations of called procedures.

$$\textsc{MatchCall}(q \xrightarrow{\overrightarrow{\mathbf{z}'} = call_j(\overrightarrow{\mathbf{t}})} q' \in \Delta_i, \nu \in \mathcal{V}_i, \nu_1 \in \mathcal{V}_j) \stackrel{def}{\equiv}$$
$$\bigwedge \begin{cases} \nu(x) = \nu_1(x) \text{ for all } x \in \mathbf{x}_g & \text{(values of global variables match)} \\ \nu((\overrightarrow{\mathbf{t}})_k) = \nu_1((\overrightarrow{\mathbf{x}}_i^{in})_k) \text{ for all } 1 \leq k \leq |\overrightarrow{\mathbf{t}}| & \text{(input values match)} \end{cases}$$

$$\textsc{MatchCallRet}(q \xrightarrow{\overrightarrow{\mathbf{z}'} = call_j(\overrightarrow{\mathbf{t}})} q' \in \Delta_i, \nu, \nu' \in \mathcal{V}_i, \nu_1, \nu_2 \in \mathcal{V}_j) \stackrel{def}{\equiv}$$
$$\bigwedge \begin{cases} \textsc{MatchCall}(q \xrightarrow{\overrightarrow{\mathbf{z}'} = call_j(\overrightarrow{\mathbf{t}})} q', \nu, \nu_1) & \\ \nu'(x) = \nu_2(x) \text{ for all } x \in \mathbf{x}_g & \text{(values of global variables match)} \\ \nu'((\overrightarrow{\mathbf{z}})_k) = \nu_2((\overrightarrow{\mathbf{x}}_i^{out})_k) \text{ for all } 1 \leq k \leq |\overrightarrow{\mathbf{z}}| & \text{(output values match)} \\ \nu(x) = \nu'(x) \text{ for all } x \in \mathbf{x}_i \setminus \overrightarrow{\mathbf{z}} & \text{(frame rule)} \end{cases}$$

Informally, MATCHCALL evaluates to true if a valuation ν at a call site of a procedure P_i is compatible with a valuation ν_1 of a called procedure P_j. MATCHCALLRET

moreover requires that a valuation ν' at a return site of a procedure P_i is compatible with a valuation ν_2 of a called procedure P_j and that the frame rule is respected.

For each procedure $P_i = \langle \mathbf{x}_i, \overrightarrow{\mathbf{x}}_i^{in}, \overrightarrow{\mathbf{x}}_i^{out}, Q_i, q_{0,i}, q_{f,i}, q_{e,i}, \Delta_i \rangle$, we define the set of summaries compatible with P_i as $\mathcal{S}_i = \{S_i : Q_i \times Q_i \to \mathcal{V}_i \times \mathcal{V}_i\}$. Intuitively, a summary S_i of a procedure P_i maps each pair of control states $(q, q') \in Q_i \times Q_i$ to a relation $S_i(q, q') \in \mathcal{V}_i \times \mathcal{V}_i$ between valuations in q and q' that are feasible by an execution of P_i that starts in q and ends in q'. For a program $\mathcal{P} = \langle \mathbf{x}_g, \{P_1, \ldots, P_n\}, P_m \rangle$, the set of summaries compatible with $\mathcal{P}$ is defined as $\mathcal{S} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_n$.

Given two configurations $\langle q, \nu \rangle$ and $\langle q', \nu' \rangle$ of a procedure P_i, the configuration $\langle q', \nu' \rangle$ is said to be an *immediate successor* of $\langle q, \nu \rangle$, with respect to a program summary $S = (S_1, \ldots, S_n) \in \mathcal{S}$, if and only if either:

- $q \xrightarrow{R} q' \in \Delta_i$ and $\nu, \nu' \models R$ (internal action)
- $q \xrightarrow{\overrightarrow{\mathbf{z}} = call_j(\overrightarrow{\mathbf{t}})} q' \in \Delta_i$ and MATCHCALLRET($q \xrightarrow{\overrightarrow{\mathbf{z}} = call_j(\overrightarrow{\mathbf{t}})} q', \nu, \nu', \nu_1, \nu_2$) for some $(\nu_1, \nu_2) \in S_j(q_{0,j}, q_{f,j})$ (successful call)
- $q' = q_{e,i}$, $q \xrightarrow{\overrightarrow{\mathbf{z}} = call_j(\overrightarrow{\mathbf{t}})} q'' \in \Delta_i$ and MATCHCALLRET($q \xrightarrow{\overrightarrow{\mathbf{z}} = call_j(\overrightarrow{\mathbf{t}})} q''$, ν, ν', ν_1, ν_2) for some $q'' \in Q_i$ and for some $(\nu_1, \nu_2) \in S_j(q_{0,j}, q_{e,j})$ (erroneous call)

A *run* of length k of a procedure P_i from q to q', under a program summary $S \in \mathcal{S}$, is a finite sequence $\langle q_0, \nu_0 \rangle \to \langle q_1, \nu_1 \rangle \to \ldots \to \langle q_k, \nu_k \rangle$, such that $q = q_0$, $q' = q_k$, and $\langle q_{i+1}, \nu_{i+1} \rangle$ is an immediate successor of $\langle q_i, \nu_i \rangle$ with respect to S, for all $0 \leq i < k$.

The *summary of a procedure* P_i under a program summary $S \in \mathcal{S}$ is a mapping $[\![P_i]\!]_S \in \mathcal{S}_i$ defined for each $q, q' \in Q_i$ as

$$[\![P_i]\!]_S(q, q') \stackrel{def}{=} \{(\nu, \nu') \mid (q, \nu) \to \ldots \to (q', \nu') \text{ is a run of } P_i \text{ of length } k \geq 1 \text{ under } S\}$$

The *summary of a program* $\mathcal{P}$, denoted by $[\![\mathcal{P}]\!]$, is defined as the least fixpoint of the function

$$\begin{array}{rcl} \mathcal{S} & \to & \mathcal{S} \\ S \in \mathcal{S} & \mapsto & ([\![P_1]\!]_S, \ldots, [\![P_n]\!]_S) \end{array}$$

We denote the components of the program summary $[\![\mathcal{P}]\!]$ as $[\![\mathcal{P}]\!] = ([\![P_1]\!], \ldots, [\![P_n]\!])$. Intuitively, $[\![P_i]\!]$ represents the reachability relation between two arbitrary control states of P_i in a finite and non-zero number of steps.

Further, we write $[\![\mathcal{P}]\!]^f = ([\![P_1]\!]^f, \ldots, [\![P_n]\!]^f)$, where $[\![P_i]\!]^f \stackrel{def}{=} [\![P_i]\!](q_{0,i}, q_{f,i})$, to denote only the summaries from the initial to the final control state. Further, we define the *final state summary* as $[\![\mathcal{P}]\!]^f_\exists = ([\![P_1]\!]^f_\exists, \ldots, [\![P_n]\!]^f_\exists)$ where

$$[\![P_i]\!]^f_\exists \stackrel{def}{=} \exists(\mathbf{x}_i \cup \mathbf{x}'_i) \setminus (\overrightarrow{\mathbf{x}}_i^{in} \cup \overrightarrow{\mathbf{x}}_i'^{out}) \,.\, [\![P_i]\!]^f$$

Intuitively, $[\![\mathcal{P}]\!]^f_\exists$ is computed from $[\![\mathcal{P}]\!]^f$ by eliminating variables that are not in the signature of P_i. Similarly, we define $[\![\mathcal{P}]\!]^e = ([\![P_1]\!]^e, \ldots, [\![P_n]\!]^e)$ and the *error state summary* $[\![\mathcal{P}]\!]^e_\exists = ([\![P_1]\!]^e_\exists, \ldots, [\![P_n]\!]^e_\exists)$, where $[\![P_i]\!]^e \stackrel{def}{=} [\![P_i]\!](q_{0,i}, q_{e,i})$ and

$$[\![P_i]\!]^e_\exists \quad \stackrel{def}{=} \quad \exists(\mathbf{x}_i \cup \mathbf{x}'_i) \setminus (\overrightarrow{\mathbf{x}}^{in}_i \cup \overrightarrow{\mathbf{x}}'^{out}_i) \,.\, [\![P_i]\!]^e$$

4.3. Computing Program Summaries

As we deal only with non-recursive programs in this paper, the call graph of a program $\mathcal{P} = \langle \mathbf{x}_g, \{P_1, \ldots, P_n\}, P_m \rangle$, denoted $CG(\mathcal{P})$ is a dag, and therefore one can choose a topological ordering of the procedures $P_{i_1}, \ldots, P_{i_n}$, such that for all $1 \leq k < \ell \leq n$, there is no path from P_{i_k} to P_{i_ℓ} in $CG(\mathcal{P})$.

Algorithm 1 computes Presburger formulae defining the summaries of the procedures, in the given topological order, starting with the procedures that do not have calls to other procedures. Since the program is not recursive, the fixpoint $[\![\mathcal{P}]\!]$ is reached in at most n steps because each procedure needs to be evaluated only once. Once the summary of a procedure is computed, it is used to replace the call transitions involving that procedure in every procedure which is higher in the topological order w.r.t.

Algorithm 1 Program Summary Algorithm

---

input a program $\mathcal{P} = \langle \mathbf{x}_g, \{P_{i_1}, \ldots, P_{i_n}\}, P_m \rangle$ ordered w.r.t. $CG(\mathcal{P})$
output a summary $([\![\mathcal{P}]\!]^f_\exists, [\![\mathcal{P}]\!]^e_\exists)$ of the program

1: **function** PROGRAMSUMMARY($\mathcal{P} = \langle \mathbf{x}_g, \{P_{i_1}, \ldots, P_{i_n}\}, P_m \rangle$)
2: $([\![\mathcal{P}]\!]^f_\exists, [\![\mathcal{P}]\!]^e_\exists) \leftarrow (\langle \emptyset, \ldots, \emptyset \rangle, \langle \emptyset, \ldots, \emptyset \rangle)$
3: **for** $k = 1, \ldots, n$ **do**
4: $([\![P_{i_k}]\!]^f_\exists, [\![P_{i_k}]\!]^e_\exists) \leftarrow$ PROCSUMMARY$(P_{i_k}, [\![\mathcal{P}]\!]^f_\exists, [\![\mathcal{P}]\!]^e_\exists)$
5: **if** V^e_m is satisfiable **then**
6: **report** "program is unsafe"

input a procedure $P = \langle \mathbf{x}, \overrightarrow{\mathbf{x}}^{in}, \overrightarrow{\mathbf{x}}^{out}, Q, q_0, q_f, q_e, \Delta \rangle$, and
a program summary $([\![\mathcal{P}]\!]^f_\exists, [\![\mathcal{P}]\!]^e_\exists)$
output a summary of P w.r.t. $(\mathcal{V}_f, \mathcal{V}_e)$

1: **function** PROCSUMMARY(P, $(\mathcal{V}_f, \mathcal{V}_e)$)
2: **for each** $q \xrightarrow{\overrightarrow{\mathbf{z}}' = call_j(\overrightarrow{\mathbf{t}})} q' \in \Delta$ **do**
3: $R_f \leftarrow$ PLUGSUMMARY$(i, j, [\![\mathcal{P}_i]\!]^f_\exists, \overrightarrow{\mathbf{t}}, \overrightarrow{\mathbf{z}}')$
4: $R_e \leftarrow$ PLUGSUMMARY$(i, j, [\![\mathcal{P}_i]\!]^e_\exists, \overrightarrow{\mathbf{t}}, \overrightarrow{\mathbf{z}}')$
5: $\Delta \leftarrow \big(\Delta \setminus \{q \xrightarrow{\overrightarrow{\mathbf{z}}' = call_j(\overrightarrow{\mathbf{t}})} q'\}\big) \cup \{q \xrightarrow{R_f} q',\ q \xrightarrow{R_e} q_e\}$
6: **return** PROCSUMMARYNOCALLS (P)

1: **function** PLUGSUMMARY($i, j, R, \overrightarrow{\mathbf{t}}, \overrightarrow{\mathbf{z}}'$)
2: **return** $R\big[(\overrightarrow{\mathbf{t}})_k / (\overrightarrow{\mathbf{x}}^{in}_i)_k\big]_{k=1..|\overrightarrow{\mathbf{t}}|} \big[(\overrightarrow{\mathbf{z}})'_k / (\overrightarrow{\mathbf{x}}^{out}_i)'_k\big]_{k=1..|\overrightarrow{\mathbf{z}}|} \wedge Id_{(\mathbf{x}_i \setminus \overrightarrow{\mathbf{x}}^{out}_i)}$

Algorithm 2 Procedure Summary Algorithm

input a procedure $P = \langle \mathbf{x}, \overrightarrow{\mathbf{x}}^{in}, \overrightarrow{\mathbf{x}}^{out}, Q, q_0, q_f, q_e, \Delta \rangle$ without call transitions
output The summary relations $([\![P]\!]^f_\exists, [\![P]\!]^e_\exists)$

1: **function** PROCSUMMARYNOCALLS(P)
2: changed $\leftarrow$ true
3: **while** changed **do**
4: changed $\leftarrow$ false
5: **for each** $q \in Q \setminus \{q_0, q_f, q_e\}$ with self-loops $q \xrightarrow{R_1} q, \ldots, q \xrightarrow{R_k} q$ **do**
6: $T \leftarrow$ DISJTRANSITIVECLOSURE$(R_1, \ldots, R_k)$
7: **for each** $q_1 \xrightarrow{P} q$ and $q \xrightarrow{Q} q_2$ such that $q_1 \neq q,\ q_2 \neq q$ **do**
8: **if** $\forall q_1 \xrightarrow{R'} q_2 \in \Delta \,.\, P \circ T \circ Q \not\Rightarrow R'$ **then**
9: $\Delta \leftarrow \Delta \cup \{q_1 \xrightarrow{P \circ T \circ Q} q_2\}$
10: changed $\leftarrow$ true
11: $Q \leftarrow Q \setminus \{q\}$
12: $\Delta \leftarrow \Delta \setminus \big(\{q \xrightarrow{R_1} q, \ldots, q \xrightarrow{R_k} q\} \cup \{q' \xrightarrow{R'} q, q \xrightarrow{R''} q'' \mid q', q'' \in Q\}\big)$
13: **return** $\big($PROJECT$(\bigvee\{R \mid q_0 \xrightarrow{R} q_f\}),$ PROJECT$(\bigvee\{R \mid q_i \xrightarrow{R} q_e\})\big)$

1: **function** PROJECT(R)
2: **return** $\exists(\mathbf{x}_i \cup \mathbf{x}'_i) \setminus (\overrightarrow{\mathbf{x}}^{in}_i \cup \overrightarrow{\mathbf{x}}'^{out}_i) \,.\, R$

$CG(\mathcal{P})$ (function PROCSUMMARY, lines 3 and 4). Additionally, the algorithm checks for error traces by also computing the error summary of each procedure and checking the resulting formula for satisfiability. The problem is thus reduced to computing the summary of a procedure without call transitions.

Algorithm 2 implements the function PROCSUMMARYNOCALLS, used to generate the (error) summary of a procedure without call transitions. The idea of this algorithm is to eliminate control states which are neither initial, error, or final, while introducing new transitions labeled with compositions of relations between the remaining states. We iterate the following until no more states can be eliminated. For each control state with (possibly zero) self-loops labeled with relations $R_1, \ldots, R_k$, we compute the transitive closure $T = (R_1 \vee \ldots \vee R_k)^*$. By convention, if $k = 0$, then T is the identity relation. Then we compose the relation of each incoming transition $q_1 \xrightarrow{R} q$ with T and with the relation of each outgoing transition $q \xrightarrow{Q} q_2$ and replace the pair of incoming and outgoing transitions with the transition $q_1 \xrightarrow{P \circ T \circ Q} q_2$, which avoids q. Finally, we eliminate q and all transitions involving it from the procedure. For an example of a run of this algorithm, the reader may refer to Figure 2 (a), (b), (c), (f), and (g).

The termination of Algorithm 2 is clearly determined by the termination of the transitive closure computation function DISJTRANSITIVECLOSURE since at each iter-

ation of the main loop, the boolean variable `changed` is set to true only if at least one control state is eliminated. Since the set of control states is finite, this implies the termination of the main loop provided that the semi-algorithm DISJTRANSITIVECLOSURE terminates.

4.4. Computing Transitive Closures of Disjunctive Relations

In this section, we present a semi-algorithm for computing transitive closures of disjunctive relations. Given a (possibly empty) set of Presburger definable relations $\{R_1, \ldots, R_k\}$, Algorithm 3, if it terminates, returns the reflexive and transitive closure of the disjunctive relation $R = R_1 \vee \ldots \vee R_k$. We assume in the following that the transitive closure R_i^+ of each relation in the set is Presburger definable. The algorithm enumerates all unrollings of the relations $R_1, \ldots, R_k$ and computes increasingly larger underapproximations of R^+. If two successive such underapproximations are equivalent, the algorithm terminates and returns the precise transitive closure.

More precisely, Algorithm 3 builds breadth-first a tree structure in which each edge corresponds to a relation R_i, and each node corresponds to the composition of the transitive closures of all relations along the path from the root to itself. The algorithm backtracks either when the composition becomes unsatisfiable, or when it is included in the union of the relations corresponding to the nodes which have been already constructed (i.e. if the test on line 7 fails). As an optimization, if a node N is obtained from its parent by composing it with the transitive closure R_i^+, it is not necessary to add the child corresponding to R_i to N since this child would be automatically subsumed by N (line 9). The result of the algorithm is the disjunction of all relations corresponding to nodes in the tree (line 11). For an example of a terminating execution of this algorithm, the reader may refer to Figure 2 (d) and (e).

Algorithm 3 Reflexive and Transitive Closure of Disjunctive Relations

input a set of relations $S = \{R_1, \ldots, R_k\}$
output The reflexive and transitive closure $(R_1 \vee \ldots \vee R_k)^*$
$Queue, Tree \leftarrow \emptyset$

```
1:  function DISJTRANSITIVECLOSURE(S)
2:      if S = ∅ then
3:          return Id
4:      Queue.add(Id, ⊥)
5:      while !Queue.empty() do
6:          ⟨N, P⟩ ← Queue.remove()
7:          if N is satisfiable and N ⇏ ⋁Tree then
8:              Tree ← Tree ∪ {N}
9:              for each R ∈ S such that R ≠ P do
10:                 Queue.add(N ∘ R+, R)
11:     return ⋁Tree
```

Theorem 1. *Let $S = \{R_1, \ldots, R_k\}$ be the input to the Algorithm 3. If the algorithm terminates, then the output is $(R_1 \vee \ldots \vee R_k)^*$.*

Proof: The reflexive and transitive closure of a relation $R = R_1 \vee \ldots \vee R_k$ is the limit of the increasing sequence defined by $S_0 = \bot$, and $S_{i+1} = S_i \vee (S_i \circ R)$, i.e. $R^* = \bigvee_{i=0}^{\infty} S_i$. Let $T_0, T_1, T_2 \ldots$ be the sequence of trees, as generated by the algorithm. Clearly, $T_0 \subseteq T_1 \subseteq T_2 \subseteq \ldots$, since the algorithm only adds new nodes, but never erases existing nodes. We will show that:

1. for each $i \geq 0$, there exists $j \geq 0$, such that $S_i \Rightarrow \bigvee T_j$
2. $\bigvee T_j \Rightarrow R^*$ for all $j \geq 0$

If the algorithm terminates, the sequence $T_0, T_1, T_2, \ldots$ has a maximal element T, and since, by (1), for all $i \geq 0$, $S_i \Rightarrow \bigvee T$, we have that $R^* \Rightarrow \bigvee T$, and therefore $\bigvee T = R^*$, by (2).

To show (1), let $\overrightarrow{\mathbf{s}} = R_{i_1} \circ \ldots \circ R_{i_t}$ be a maximal satisfiable sequence of compositions, in some iteration S_i of the sequence leading to R^*. We will exhibit a tree $T_{\overrightarrow{\mathbf{s}}}$ produced by the algorithm, such that $\overrightarrow{\mathbf{s}} \Rightarrow \bigvee T_{\overrightarrow{\mathbf{s}}}$. Since S_i has finitely many compositions $\overrightarrow{\mathbf{s}}$, and since the $\{T_i\}_{i \geq 0}$ sequence is strictly increasing, it is enough to take T_j as the maximal tree $T_{\overrightarrow{\mathbf{s}}}$ in the sequence.

Let $i_{k_1}, \ldots, i_{k_s}$ be the subsequence of $i_1, \ldots, i_t$ obtained by replacing each (stuttering) block of consisting of repetitions of the same relation R_{i_j} by R_{i_j}. Since $R_{i_1} \circ \ldots \circ R_{i_t}$ is satisfiable, then $R_{i_1}^+ \circ \ldots \circ R_{i_t}^+$ is satisfiable as well. Notice that every prefix of this sequence is satisfiable. Let $R_{i_1}^+ \circ \ldots \circ R_{i_m}^+$ be the maximal prefix of $\overrightarrow{\mathbf{s}}$ for which there exists some tree T_ℓ, which contains the relation corresponding to this prefix at the tree position $i_1 \cdot \ldots \cdot i_m$. Clearly $R_{i_1}^+ \circ \ldots \circ R_{i_m}^+ \Rightarrow \bigvee T_\ell$. To find the tree which subsumes the entire sequence, we iterate the following, for $h = m+1, \ldots, t$:

- if $R_{i_1}^+ \circ \ldots \circ R_{i_h}^+ \not\Rightarrow \bigvee T_\ell$, let Q be the smallest prefix-closed subset of positions in T_ℓ which subsumes $R_{i_1}^+ \circ \ldots \circ R_{i_{h-1}}^+$.
- let T' be the tree obtained from T_ℓ by adding to each maximal position p in Q, a node $\langle p \cdot i_h, T_\ell(p) \circ R_{i_h}^+ \rangle$, only if $T_\ell(p) \circ R_{i_h}^+ \not\Rightarrow \bigvee T_\ell$.
- let T_ℓ be the next tree in the sequence which contains T'.

To prove (2), observe that the summary corresponding to each path in in some tree T_i generated by the algorithm is subsumed by R^*, hence $\bigvee_{i \geq 0} T_i \Rightarrow R^*$. □

As previously mentioned, Algorithm 1 computing a program summary terminates for non-recursive programs provided that each call to the DISJTRANSITIVECLOSURE function terminates. In general, this is, however, not true due to the undecidability of the safety problem for integer programs [44].

4.4.1. Over- and Underapproximations of Transitive Closures

Termination of the disjunctive closure algorithm can be imposed in two ways, depending on the goal of the analysis. If the goal is proving correctness of the system, i.e. unreachability of the error states, one may resort to overapproximation of the disjunctive loop relations $R_1 \vee \ldots \vee R_k$ by a single, weaker, relation $R^\sharp$ (i.e. $R_1 \vee \ldots \vee R_k \Rightarrow R^\sharp$) whose reflexive and transitive closure is Presburger definable and can be computed by our methods. If unreachability of error states in the program can be proved using the overapproximation instead of the original relation, then this

constitutes a valid correctness proof, since every trace of the original program is a trace of the abstract program.

To this end, Algorithm 3 can be modified into Algorithm 4, which guarantees termination, at the expense of giving up precision. The crux of Algorithm 4 is that the expansion of a tree node situated at a depth beyond a certain threshold is done by composition with $R^{\sharp +}$ (line 14), instead of R_i^+ for some $i = 1, \ldots, k$ as in the case of nodes below the threshold (line 11). Since $R_1 \vee \ldots \vee R_k \Rightarrow R^\sharp$, then for any sequence $i_1, \ldots, i_m \in \{1, \ldots, k\}$, the relation $R_{i_1}^+ \circ \ldots \circ R_{i_m}^+$ is subsumed by $R^{\sharp +}$. This prevents future descendants of nodes above the threshold to be added to the queue (the test on line 7 will fail for them), causing termination of the main loop. The returned value in this case is an overapproximation of the reflexive and transitive closure $(R_1 \vee \ldots \vee R_k)^*$.

Algorithm 4 Abstract Reflexive and Transitive Closure of Disjunctive Relations

input a set of relations $S = \{R_1, \ldots, R_k\}$
output a relation R such that $(R_1 \vee \ldots \vee R_k)^* \Rightarrow R$
$Queue, Tree \leftarrow \emptyset$

1: **function** ABSTRANSITIVECLOSURE(S)
2: **if** $S = \emptyset$ **then**
3: **return** Id
4: $Queue.add(Id, \bot, 0)$
5: **while** !$Queue.empty()$ **do**
6: $\langle N, P, depth \rangle \leftarrow Queue.remove()$
7: **if** N is satisfiable and $N \not\Rightarrow \bigvee Tree$ **then**
8: $Tree \leftarrow Tree \cup \{N\}$
9: **if** $depth \leq Threshold$ **then**
10: **for each** $R \in S$ such that $R \neq P$ **do**
11: $Queue.add(N \circ R^+, R, depth + 1)$
12: **else**
13: $R^\sharp \leftarrow$ ABSTRACTDISJRELATION(S)
14: $Queue.add(N \circ R^{\sharp +}, R, depth + 1)$
15: **return** $\bigvee Tree$

On the other hand, if the goal of the analysis is to find errors in the program, the disjunctive closure algorithm can be stopped after a given number of steps, the result being an underapproximation of the transitive closure, i.e. a relation $R^\flat \Rightarrow (R_1 \vee \ldots \vee R_k)^*$. Even if they cannot be used to certify correctness of systems, underapproximations play an important role in finding errors within complex systems since every error trace found using underapproximations is a valid error trace of the program.

A direct consequence of the proof of Theorem 1 is that the disjunction of the nodes in each tree built by the DISJTRANSITIVECLOSURE function is an underapproximation of the transitive closure. Thus the algorithm can be stopped if, e.g., the number of nodes reaches a predefined threshold, and the result will be a relation stronger than $(R_1 \vee \ldots \vee R_k)^*$. This relation can be used in Algorithm 2 to underapproximate the

summary of a procedure and thus the summary of a program. If insufficient, the under-approximation can be improved by letting the DISJTRANSITIVECLOSURE algorithm run longer.

5. Computing Transitive Closures of Periodic Relations

In this section, we present a general framework for computing *closed forms* and *transitive closures* of certain relations, called *periodic*. The closed form of a relation $R(\mathbf{x}, \mathbf{x}')$ is a relation $\widehat{R}(k, \mathbf{x}, \mathbf{x}')$ where substituting the parameter k with an integer m gives a relation equivalent to R^m for each $m \geq 0$. Once the closed form is computed, the transitive closure of R can be defined as $\exists k \geq 1 \,.\, \widehat{R}(k, \mathbf{x}, \mathbf{x}')$.

We define a notion of periodicity on classes of relations that can be naturally represented as matrices. In general, an infinite sequence of integers is said to be *periodic* if the elements of the sequence situated at equal distance one from another differ by the same quantity (while admitting some non-periodic initial prefix of finite length in the sequence). This definition is generalized to matrices of integers, entry-wise. Assuming that each finite power R^k of a relation R is represented by a matrix M_k, R is said to be periodic if the infinite sequence $\{M_k\}_{k=0}^{\infty}$ of matrix representations of powers of R is periodic.

Periodicity guarantees that the sequence has an infinite subsequence which can be captured by an existentially quantified formula, which defines infinitely many powers of the relation. Then, the remaining powers can be computed by composing the existentially quantified formula with only finitely many (given by the size of the period) powers of the relation.

For instance, consider the relation R defined as $x' = y + 1 \wedge y' = x$. This relation is periodic, and its odd powers $R^1, R^3, R^5, \ldots$ can be represented by the formula $\exists \ell \geq 0 \,.\, (x' = y + \ell + 1 \wedge y' = x + \ell)$. Even powers $R^2, R^4, R^6, \ldots$ can then be represented by composing this formula with R.

We show that the closed form of a periodic relation can be defined, once the *period* and the *prefix* of the relation are known. We present a generic algorithm that finds a period and a prefix of periodic relations and computes their closed form and transitive closure.

5.1. Periodic Sequences

We first define the notion of periodic sequences of integers. This definition can be generalized to arbitrary semirings.

Definition 1. *Given an infinite sequence $\{s_k\}_{k=0}^{\infty} \in \mathbb{Z}_\infty$ of integers, we say that it is* periodic *if and only if there exist integers $b \geq 0$, $c > 0$ and $\lambda_0, \ldots, \lambda_{c-1} \in \mathbb{Z}_\infty$ such that $s_{b+(k+1)c+i} = \lambda_i + s_{b+kc+i}$, for all $k \geq 0$ and $i = 0, 1, \ldots, c-1$ The smallest values $b \in \mathbb{N}$, $c \in \mathbb{N}_+$ for which the above holds are called the* prefix *and the* period *of $\{s_k\}_{k=0}^{\infty}$. The values $\lambda_0, \lambda_1, \ldots, \lambda_{c-1} \in \mathbb{Z}_\infty$ are called the* rates *of $\{s_k\}_{k=0}^{\infty}$.*

Example 1. *The sequence $\{\sigma_k\}_{k=0}^{\infty}$ where $\sigma_0 = \sigma_1 = 10$, $\sigma_k = 5\ell + 3$ for each $k = 2\ell$, $\ell \geq 1$, and $\sigma_k = 3\ell + 1$ for each $k = 2\ell + 1$, $\ell \geq 1$, is periodic with prefix $b = 2$, period $c = 2$ and rates $\lambda_0 = 5$, $\lambda_1 = 3$. The sequence $\{\tau_k\}_{k=0}^{\infty}$ where*

$\sigma_k = 7\ell + 1$ *for each* $k = 3\ell$, $\ell \geq 0$, $\sigma_k = \ell^2$ *for each* $k = 3\ell + 1$, $\ell \geq 0$, *and* $\sigma_k = \ell^3$ *for each* $k = 3\ell + 2$, $\ell \geq 0$ *is not periodic.* □

The notion of periodic sequences extends to sequences of integer matrices:

Definition 2. *A sequence of integer matrices* $\{A_k\}_{k=0}^{\infty} \in \mathbb{Z}_{\infty}^{m \times m}$ *is said to be periodic if, for all* $1 \leq i, j \leq m$, *the sequence* $\{(A_k)_{ij}\}_{k=0}^{\infty}$ *is periodic.*

The following lemma gives an alternative characterization of periodic sequences of matrices:

Lemma 1. *An infinite sequence of matrices* $\{A_k\}_{k=1}^{\infty} \in \mathbb{Z}_{\infty}^{m \times m}$ *is periodic if and only if there exist integers* $b \geq 0$, $c > 0$ *and* $\Lambda_0, \ldots, \Lambda_{c-1} \in \mathbb{Z}_{\infty}^{m \times m}$ *such that* $A_{b+(k+1)c+i} = \Lambda_i + A_{b+kc+i}$, *for all* $k \geq 0$ *and* $i = 0, 1, \ldots, c-1$

Proof: According to the definition, $\{A_k\}_{k=1}^{\infty}$ is periodic if and only if, for each $1 \leq i, j \leq m$ there exist $b_{ij} \geq 0$, $c_{ij} > 0$ and $\lambda_l^{ij} \in \mathbb{Z}_{\infty}$ such that $\left(A_{b_{ij}+(k+1)c_{ij}+l}\right)_{ij} = \lambda_l^{ij} + \left(A_{b_{ij}+kc_{ij}+l}\right)_{ij}$ for all $k \geq 0$, $l = 0, 1, \ldots, c_{ij} - 1$. Let c be the least common multiple of all c_{ij}, b be the maximum of all b_{ij} and let Λ_t, $t = 0, 1, \ldots, c-1$ be the matrix defined as:

$$(\Lambda_t)_{ij} = \left(\lambda^{ij}_{(b-b_{ij}+t) \mod c_{ij}}\right) \cdot \frac{c}{c_{ij}}$$

The condition $A_{b+(k+1)c+i} = \Lambda_i + A_{b+kc+i}$ is verified for all $k \geq 0$ and $i = 0, 1, \ldots, c-1$, with the above definitions. □

5.2. Periodic Relations

Let $\mathbf{x} = \{x_1, \ldots, x_N\}$ be a set of variables. In this section, we consider that $\mathcal{R}$ is a class of first-order arithmetic formulae with free variables in $\mathbf{x} \cup \mathbf{x}'$. These formulae denote integer relations $R \subseteq \mathbb{Z}^N \times \mathbb{Z}^N$. We define $*$-*consistent* and periodic relations.

Definition 3. *A relation* R *is* $*$*-consistent if and only if* R^n *is consistent for all* $n \geq 0$.

Definition 4. *A relation* $R \in \mathcal{R}$ *is said to be* periodic *if and only if, either (1) it is not* $*$*-consistent, or (2) it is* $*$*-consistent and there exist two functions:*

- $\sigma : \mathcal{R} \to (\mathbb{Z}_{\infty}^{m \times m})_{\perp}$ *mapping each* consistent *relation from* $\mathcal{R}$ *into a matrix, and each inconsistent relation into* $\perp$
- $\rho : \mathbb{Z}_{\infty}^{m \times m} \to \mathcal{R}$ *mapping each matrix into a relation from* $\mathcal{R}$ *such that* $\rho(\sigma(R)) \Leftrightarrow R$ *for each consistent relation* $R \in \mathcal{R}$

such that the infinite sequence of matrices $\{\sigma(R^i)\}_{i=0}^{\infty} \in \mathbb{Z}_{\infty}^{m \times m}$ *is periodic.*

If each relation $R \in \mathcal{R}$ is periodic, then the class of relations $\mathcal{R}$ is called periodic as well. The following lemma gives an alternative characterization of $*$-consistent periodic relations.

Lemma 2. *A $*$-consistent relation R is periodic if and only if there exist integers $b \geq 0$, $c > 0$, $m > 0$ and a matrices $\Lambda_0, \ldots, \Lambda_{c-1} \in \mathbb{Z}_\infty^{m \times m}$ such that*

$$R^{nc+b+i} \Leftrightarrow \rho(n \cdot \Lambda_i + \sigma(R^{b+i}))$$

for all $n \geq 0$ and for all $0 \leq i < c$.

Proof: By induction on $n \geq 0$, we prove that $\sigma(R^{nc+b+i}) = n \cdot \Lambda + \sigma(R^{b+i})$, for all $n \geq 0$ and for all $0 \leq i < c$. The base case trivially holds. For the induction step, observe that

$$\sigma(R^{b+i+(n+1)c}) = \Lambda_i + \sigma(R^{b+i+nc}) = \Lambda + n \cdot \Lambda + \sigma(R^{b+i}) = (n+1) \cdot \Lambda_i + \sigma(R^b).$$

The first equality is by Lemma 1, the second is by the induction hypothesis. □

Next, we define prefix, period and rates of periodic relations.

Definition 5. *If R is a $*$-consistent and periodic relation, we call* prefix, period *and* rates *of R the minimal integers $b \geq 0$, $c > 0$ and matrices $\Lambda_0 \ldots, \Lambda_{c-1} \in \mathbb{Z}_\infty^{m \times m}$ satisfying the condition of Lemma 2. If R is not $*$-consistent, we define its prefix and period as $b = \inf\{n \geq 0 \mid R^n$ is inconsistent $\}$ and $c = 1$.*

If k is a variable not in $\mathbf{x} \cup \mathbf{x}'$, let $\mathcal{R}[k]$ be the class of first-order arithmetic formulae with free variables from the set $\mathbf{x} \cup \mathbf{x}' \cup \{k\}$. The variable k is called a parameter, and these formulae are called *parametric relations*, in the following. The following definition emphasizes the role of parametric relations.

Definition 6. *Let $R(\mathbf{x}, \mathbf{x}') \in \mathcal{R}$ be a relation. The* closed form *of R is the formula $\widehat{R}(k, \mathbf{x}, \mathbf{x}') \in \mathcal{R}[k]$ such that $\widehat{R}(k, \mathbf{x}, \mathbf{x}')[n/k] \Leftrightarrow R^n(\mathbf{x}, \mathbf{x}')$, for all $n \geq 0$.*

It follows immediately from the above definition that the closed form of a relation is unique, up to equivalence. Defining the transitive closure of a relation is closely related to defining its closed form, since $R^+(\mathbf{x}, \mathbf{x}') \Leftrightarrow \exists k > 0 \,.\, \widehat{R}(k, \mathbf{x}, \mathbf{x}')$, for all $R \in \mathcal{R}$.

The algorithm presented in this section computes the transitive closure of a periodic relation R by computing the closed form of a subsequence $\{\sigma(R^{b+nc})\}_{n \geq 0}$ for some $b \geq 0, c > 0$ (not necessarily the prefix and the period of R).

Definition 7. *Given a relation $R(\mathbf{x}, \mathbf{x}')$ and integers $b \geq 0$, $c > 0$, the* closed form *of the the infinite sequence $\{R^{b+nc}\}_{n \geq 0}$ is a formula $\widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}')$ such that*

$$\widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}')[n/\ell] \Leftrightarrow R^{b+nc}$$

for all $n \geq 0$.

Once the closed form $\widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}')$ is computed for some $b \geq 0, c > 0$, both the transitive closure and the closed form of R can be defined as shown by the following lemma.

Lemma 3. *Let R be a relation, $b > 0$, $c > 0$ be arbitrary integers and $\widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}')$ be the closed form of the infinite sequence $\{R^{b+nc}\}_{n\geq 0}$. Then, the transitive closure of R can be defined as:*

$$R^+ \Leftrightarrow \big(\bigvee_{i=1}^{b-1} R^i\big) \ \vee \ \exists \ell \geq 0 \,.\, \widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}') \circ \big(\bigvee_{j=0}^{c-1} R^j\big)$$

and the closed form of R can be defined as

$$\widehat{R}(k, \mathbf{x}, \mathbf{x}') \Leftrightarrow \big(\bigvee_{i=1}^{b-1} (k = i) \wedge R^i\big) \vee \exists \ell \geq 0 \,.\, \widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}') \circ \big(\bigvee_{j=0}^{c-1} (k = b + \ell c + j) \wedge R^j\big)$$

Proof: Let $\widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}')$ be the closed form of $\{R^{b+nc}\}_{n\geq 0}$ for some integers $b > 0, c > 0$. Observe that

$$\begin{array}{rcl} R^+ \ \Leftrightarrow \ \bigvee_{i=1}^{\infty} R^i & \Leftrightarrow & \big(\bigvee_{i=1}^{b-1} R^i\big) \ \vee \ \big(\bigvee_{n=0}^{\infty} R^{nc+b}\big) \circ \big(\bigvee_{j=0}^{c-1} R^j\big) \\ & \Leftrightarrow & \big(\bigvee_{i=1}^{b-1} R^i\big) \ \vee \ \exists \ell \geq 0 \,.\, \widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}') \circ \big(\bigvee_{j=0}^{c-1} R^j\big) \end{array}$$

The last equivalence above follows from Definition 7. In a similar way, we infer that the closed form of a relation R can be defined as

$$\widehat{R}(k, \mathbf{x}, \mathbf{x}') \Leftrightarrow \big(\bigvee_{i=1}^{b-1} (k = i) \wedge R^i\big) \vee \exists \ell \geq 0 \,.\, \widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}') \circ \big(\bigvee_{j=0}^{c-1} (k = b + \ell c + j) \wedge R^j\big),$$

which completes the proof. □

Next, consider a function $\pi : \mathbb{Z}[k]_{\infty}^{m\times m} \rightarrow \mathcal{R}[k]$ mapping matrices of linear terms of the form $\alpha \cdot k + \beta$, with integer coefficients, into parametric relations $R(k, \mathbf{x}, \mathbf{x}')$, such that

$$\pi(M)[n/k] \Leftrightarrow \rho(M[n]), \text{for all } n \in \mathbb{Z}$$

for all $M \in \mathbb{Z}[k]_{\infty}^{m\times m}$. In other words, π is the *parametric counterpart* of the ρ function from Definition 4. Concrete examples of parametric matrix-relations mappings will be given in Sections 7.2, 7.5, and 7.8.

We next show that the transitive closure and the closed form of a periodic relation R with prefix b, period c, and rates $\Lambda_0, \ldots, \Lambda_{c-1}$ can be defined in first order arithmetic. If R is not $*$-consistent, then clearly $R^+ \Leftrightarrow \bigvee_{i=1}^{b-1} R^i$ and $\widehat{R}(k, \mathbf{x}, \mathbf{x}') \Leftrightarrow \bigvee_{i=1}^{b-1}(k = i \wedge R^i)$. If R is $*$-consistent, then by Lemma 2, $R^{nc+b} \Leftrightarrow \rho(n \cdot \Lambda_0 + \sigma(R^b))$. Then, we apply the following proposition which shows that the closed form of the sequence $\{R^{b+nc}\}_{n\geq 0}$ can be defined as $\widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}') \Leftrightarrow \pi(\ell \cdot \Lambda_0 + \sigma(R^b))$.

Proposition 1. *Let R be a relation, $b > 0, c > 0$ be arbitrary integers, and $\Lambda \in \mathbb{Z}_{\infty}^{m\times m}$ be a matrix such that, for all $n \geq 0$:*

1. *$\rho(n \cdot \Lambda + \sigma(R^b)) \not\Leftrightarrow \bot$, and*
2. *$R^{nc+b} \Leftrightarrow \rho(n \cdot \Lambda_0 + \sigma(R^b))$.*

Then, R is $*$-consistent and $\widehat{R}_{b,c}(k, \mathbf{x}, \mathbf{x}') \Leftrightarrow \pi(k \cdot \Lambda + \sigma(R^b))$.

Proof: Clearly, if both conditions hold, it follows that R is $*$-consistent. The mapping $\pi : \mathbb{Z}[k]^{m\times m}_{\infty} \to \mathcal{R}[k]$ satisfies the following equivalence for all $M \in \mathbb{Z}[k]^{m\times m}_{\infty}$ and for all $n \in \mathbb{Z}$

$$\pi(M)[n/k] \Leftrightarrow \rho(M[n])$$

Thus, letting $M = k \cdot \Lambda + \sigma(R^b)$, it follows that if the two conditions hold, then

$$\pi(M)[n/k] \Leftrightarrow \rho(M[n]) \Leftrightarrow R^{b+nc}.$$

Hence, $\pi(M) = \pi(k \cdot \Lambda + \sigma(R^b))$ is the closed form of of the sequence $\{R^{b+nc}\}_{n\geq 0}$, by Definition 7. □

Having computed $\widehat{R}_{b,c}(\ell, \mathbf{x}, \mathbf{x}')$, we can finally apply Lemma 3 to define the transitive closure and the closed form of R.

5.3. **Transitive Closure Algorithm**

The result of this section is a generic algorithm that computes the transitive closure of a given periodic relation (Algorithm 5). The algorithm needs to be instantiated for a specific class $\mathcal{R}$ of periodic relations by providing the corresponding mappings σ, ρ (Definition 4) and π (the parametric counterpart of ρ) as discussed in the previous. This algorithm can be easily adapted to compute the closed form of a relation, instead of its transitive closure, as we show in the end of this section. Next, in Section 7 we show how this algorithm can be used with three classes of relations: difference bounds, octagons, and finite monoid affine transformations.

The main idea of the algorithm is to discover integers $b \geq 0$ and $c > 0$ such that the sequence $\{\sigma(R^{b+nc})\}_{n\geq 0}$ is periodic. Provided that relation R is periodic, the enumeration on lines 2,4,16 is guaranteed to eventually find a pair (b, c) for which the algorithm terminates at line 7 or 12, as we later prove in Theorem 2. For each prefix-period candidate (b, c), we consider the first three powers supposed to be equidistant, namely R^b, R^{b+c} and R^{b+2c}, and we check that all three are consistent (lines 5-6). If at least one is inconsistent, the relation is not $*$-consistent, and the transitive closure is the disjunction of all powers up to the first one which is inconsistent (line 7). Otherwise, if R^b, R^{b+c} and R^{b+2c} are consistent, the algorithm attempts to compute the first rate of the sequence (line 8), by comparing the matrices $\sigma(R^b)$, $\sigma(R^{b+c})$ and $\sigma(R^{b+2c})$. If the distance Λ between $\sigma(R^{b+c})$ and $\sigma(R^b)$ equals the one between $\sigma(R^{b+2c})$ and $\sigma(R^{b+c})$, then Λ is a potential candidate for the rate of the sequence $\{\sigma(R^b + nc)\}_{n\geq 0}$.

The rest of the main loop (lines 8-16) is dedicated to checking whether b, c and Λ constitute a valid prefix, period and rate of the sequence of powers of R. To this end, it is sufficient to check whether (i) the sequence of relations $\{\rho(n \cdot \Lambda + \sigma(R^b))\}_{n=0}^{\infty}$ is $*$-consistent, and (ii) that it follows the pattern of a periodic sequence. The first condition is checked by the MAXCONSISTENT procedure which returns the largest $n \geq 0$ for which $rho(n \cdot \Lambda + \sigma(R^b))$ is consistent, or ∞, if no such n exists. The second condition is checked by the MAXPERIODIC procedure, which returns the largest n for which the sequence $\{\rho(n \cdot \Lambda + \sigma(R^b))\}_{n=0}^{\infty}$ is compliant with R^c, or ∞, if no such n exists.

Since, by the definition of MAXPERIODIC, either $L = K = \infty$, or else $L < K$, the following Lemma 4 ensures that if the test on line 11 succeeds for the chosen b, c and Λ, then $\pi(k \cdot \Lambda + \sigma(R^b))$ is the closed form of the sequence $\{\sigma(R^{b+nc})\}_{n\geq 0}$ and consequently, the transitive closure can be defined using Lemma 3 (line 12).

Algorithm 5 Transitive Closures of Periodic Relations

input a periodic relation R
output The transitive closure of R

1: **function** TRANSITIVECLOSURE(R)
2: **let** $P \leftarrow R$ **and** $b \leftarrow 1$ **and** $b_{jump} = 1$
3: **while** $true$ **do**
4: **for all** $c = 1, 2, \ldots, b$ **do**
5: **for all** $\ell = 0, 1, 2$ **do**
6: **if** $R^{b+\ell c} \Leftrightarrow \bot$ **then**
7: **return** $R^+ \Leftrightarrow P \vee \left(\bigvee_{i=b+1}^{b+\ell c-1} R^i \right)$
8: **if** $\exists \Lambda \,.\, \sigma(R^b) + \Lambda = \sigma(R^{b+c}) \wedge \sigma(R^{b+c}) + \Lambda = \sigma(R^{b+2c})$ **then**
9: $K \leftarrow$ MAXCONSISTENT(R, b, Λ)
10: $L \leftarrow$ MAXPERIODIC(R, b, Λ, c, K)
11: **if** $L = \infty$ **then**
12: **return** $P \vee \exists k \geq 0 \,.\, \pi(k \cdot \Lambda + \sigma(R^b)) \circ \left(\bigvee_{j=0}^{c-1} R^j \right)$
13: $b_{jump} \leftarrow \max\{b_{jump}, b + c \cdot (L+1)\}$
14: $b_{next} \leftarrow \max\{b+1, b_{jump}\}$
15: $P \leftarrow P \vee \bigvee_{i=b}^{b_{next}-1} R^i$
16: $b \leftarrow b_{next}$
17: **function** MAXCONSISTENT(R,b,Λ)
18: **return** $\sup\{n \in \mathbb{N} \mid \rho(n \cdot \Lambda + \sigma(R^b)) \not\Leftrightarrow \bot\}$
19: **function** MAXPERIODIC(R, b, Λ, c, K)
20: **return** $\sup\{n \leq K \mid \forall 0 \leq \ell < n \,.\, \rho(\ell \cdot \Lambda + \sigma(R^b)) \circ R^c \Leftrightarrow \rho((\ell+1) \cdot \Lambda + \sigma(R^b))\}$

Lemma 4. *Let R be a periodic relation, let $b > 0, c > 0$ be integers such that $R^b \not\Leftrightarrow \bot$, and $\Lambda \in \mathbb{Z}_\infty^{m \times m}$ be a matrix such that, for all $n \geq 0$:*

1. $\rho(n \cdot \Lambda + \sigma(R^b)) \not\Leftrightarrow \bot$, *and*
2. $\rho((n+1) \cdot \Lambda + \sigma(R^b)) \Leftrightarrow \rho(n \cdot \Lambda + \sigma(R^b)) \circ R^c$.

Then, R is $*$-consistent and $\widehat{R}_{b,c}(k, \mathbf{x}, \mathbf{x}') \Leftrightarrow \pi(k \cdot \Lambda + \sigma(R^b))$.

Proof: It is sufficient to prove that

$$R^{b+nc} \Leftrightarrow \rho(n \cdot \Lambda + \sigma(R^b)) \text{ for all } n \geq 0.$$

For, if the above is true, the statement of this lemma follows from Proposition 1. The base case $n = 0$ follows from Definition 4 and the fact that R^b is consistent. The induction step is as follows:

$$\begin{array}{rcll} R^{(n+1)c+b} & \Leftrightarrow & R^{nc+b} \circ R^c & \\ & \Leftrightarrow & \rho(n \cdot \Lambda + \sigma(R^b)) \circ R^c & \text{(by the induction hypothesis)} \\ & \Leftrightarrow & \rho((n+1) \cdot \Lambda + \sigma(R^b)) & \end{array}$$

□

Suppose now that the test on line 11 fails, i.e. the sequence of relations $\{\rho(n \cdot \Lambda + \sigma(R^b))\}_{n=0}^{\infty}$ is neither $*$-consistent, nor periodic. In this case we could start looking for a new prefix, period and rate, by incrementing b, setting c to one, and continuing to look for another candidate rate Λ, which satisfies the test at line 8. This could be achieved by simply incrementing b by one, i.e. without the update at line 13. However, for relations with very long prefixes, this would be quite inefficient, as shown by the following example.

Example 2. *Consider, for instance, the relation:*

$$R \equiv x' = x + 1 \wedge 0 \leq x \leq 10^9$$

The relation R^i is consistent for all $1 \leq i \leq 10^9$ and becomes inconsistent for $i = 10^9 + 1$. Without line 13, Algorithm 5 would need at least $\lceil \frac{10^9}{3} \rceil$ iterations of the main loop in order to discover the inconsistency (line 6).

Notice that MAXPERIODIC returns the span of the interval in which the relation is periodic with the current rate. The algorithm optimizes the search by storing the upper bound of the periodic interval in b_{jump}. If the sequence is periodic for none of $c = 1, \ldots, b$, line 16 updates b with the upper bound of the periodic interval in case such interval was detected, or otherwise, it increments b by one as in the unoptimized case.

Example 3. *(contd.) For the relation in the previous example, the prefix 10^9 is discovered after the first iteration, since the call to* MAXPERIODIC *with $b = c = 1$ returns $L = 10^9$. The inconsistency of the sequence $\{R^i\}_{i=1}^{\infty}$ is discovered at the second iteration of the main loop (line 6).*

For efficiency reasons, the algorithm maintains (and updates) a prefix relation P with the following invariant property:

$$P \Leftrightarrow \bigvee_{i=0}^{b} R^i \tag{1}$$

By updating P at line 15, we compute part of the prefix up to the next candidate for b.

Finally, we prove the correctness of Algorithm 5 and give bounds on the number of iterations of the main loop of Algorithm 5 and on the sizes of integers b, c considered during any iteration of the main loop, in terms of the prefix and the period of the input relation.

Theorem 2. *If R is a periodic relation with prefix B and period C, then Algorithm 5 eventually terminates after at most $(B + C)^2 + C$ iterations of the main loop at lines 4-13 and returns the transitive closure of R. Moreover, $c \leq B + C$ and $b \leq B + C + 3C^2 + 3BC$ for each prefix b and period c considered by the algorithm.*

Proof: Let us first prove that (1) holds whenever the control is at one of the lines 3-13. Initially $P \Leftrightarrow R$ and $b = 1$, so (1) holds trivially. If (1) holds before executing line 15, then it will also hold after executing lines 15-16, by the definition of MAXPERIODIC.

Let b_i (c_i, respectively), $i \geq 1$ be the value of b (c, respectively) during the i-th iteration of the main loop at line 5. We next make several observations:

1. By definition of B and C, if $b_i \geq B$ and $c_i = kC$ for some $k \geq 1$, then the algorithm returns at line 12
2. The prefix increases with each iteration of the outer loop (line 16)
3. For each considered prefix b, the algorithm consecutively tests periods in the range $1, \ldots, b$

By observation 1, proving termination of Algorithm 5 amounts to showing that $b_i \geq B$ and $c_i = kC$ for some $i, k \geq 1$. We next prove that the algorithm terminates in at most $(B + C)^2 + C$ iterations of the main loop and that $c_i \leq B + C$ and $b_i \leq B + C + 3C^2 + 3BC$ for each $i \geq 1$.

First suppose that R is not $*$-consistent and hence, $R^B \Leftrightarrow \bot$ and $R^{B-1} \not\Leftrightarrow \bot$, by definition of B. The algorithm eventually reaches line 7, since b increases with each iteration of the outer loop, by observation 2. If the test at line 8 succeeds during i-th iteration, then definition of MAXCONSISTENT procedure guarantees that $b_i + c_i K_i < B$. Since $L_i \leq K_i$ by definition of MAXPERIODIC procedure, if follows that $b_i + c_i L_i < B$. Consequently $b_{i+1} \leq B + C$. If $b_{i+1} \geq B$, the algorithm terminates in the $(i + 1)$-th iteration at line 7. The algorithm thus terminates after at most $(B - 1)^2 + 1$ iterations of the main loop. Clearly, $b_i \leq B + C$ for each $i \geq 1$ and since $c_i \leq b_i$, it follows that $c_i \leq B + C$ too.

Next, suppose that R is $*$-consistent. Let us first consider the unoptimized algorithm without line 13. Then, b is incremented by one in each iteration of the outer loop at line 16. Consequently, the algorithm returns at line 12 when $b_i = B + C$ and $c_i = C$ at the latest, by observation 3. Clearly, the algorithm terminates after at most $(B + C)^2$ iterations of the main loop and in each iteration, $c_i \leq b_i \leq B + C$.

Next, let us consider a $*$-consistent relation and the optimized algorithm with line 13. If the algorithm returns for some $b_i \leq B + C$, the bounds follow easily. Suppose that the algorithm has not returned for some $b_i \leq B + C$ and let $i \geq 1$ be the unique index such that $b_i \leq B + C$ and $b_{i+1} > B + C$. Clearly, $c_i = b_i$ and $c_{i+1} = 1$, by observation 3. Notice that if $b_{i+1} = b_i + 1$, then $b_i = B + C$ and therefore, the algorithm terminates at line 12 for prefix-period candidate $(B + C, C)$ at the latest. Thus, if the algorithm does not terminate for a prefix candidate in the range $1..(B+C)$, it cannot be that $b_{i+1} = b_i + 1$. Consequently, $b_{i+1} > b_i + 1$ and hence, $b_{i+1} = b_{jump}$ for some $b_{jump} > b_i + 1$. Since $b_{jump} > b_i + 1$, a periodic interval $\langle b_i, \ldots, b_i + Lc_i \rangle$ of the sequence $\{\sigma(R^m)\}_{m \geq 0}$ must have been detected for some $2 \leq L < \infty$ by the MAXPERIODIC procedure and thus, $b_{i+1} = b_{jump} = b_i + (L + 1)c_i$. We now demonstrate that $b_{i+1} \leq B + C + 3C^2 + 3BC$. By contradiction, suppose that $b_{i+1} > B + C + 3C^2 + 3BC$. We define (see Figure 3 for illustration):

$$Q = \mathrm{lcm}(C, c_i) \qquad d = \lceil \tfrac{B+C-b_i}{Q} \rceil \cdot \tfrac{Q}{c_i} \qquad e = b_i + c_i d \qquad j = (e - B) \bmod C$$

It follows that $e - b_i = kQ$ for some $k \geq 1$ and $e - (B + C) \leq Q$. Since $c_i \leq b_i \leq B + C$, it follows that $Q \leq C(B + C) = C^2 + BC$. Since we assume that $b_{i+1} > B + C + 3C^2 + 3BC$, it follows that $b_{i+1} - (B + C) \geq 3Q$ and therefore $b_{i+1} - e \geq 2Q$. Let $\Lambda_0, \ldots, \Lambda_{C-1}$ be the rates of the sequence $\{\sigma(R^m)\}_{m \geq 0}$ corresponding to B and C and Λ be the rate considered by the algorithm in the i-th iteration. Let denote $M_m = \sigma(R^m)$ for all $m \geq 0$. By periodicity of the sequence $\{\sigma(R^m)\}_{m \geq 0}$, it follows that $M_e + \frac{Q}{C}\Lambda_j = M_{e+Q}$. Recall that $b_{i+1} \geq e + 2Q$ and that $e - b_i$ is a multiple of c_i.

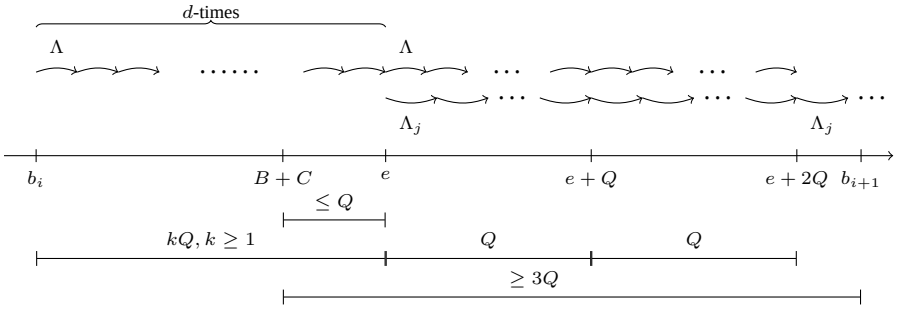


Figure 3: Bounding the prefix b_{i+1}

It follows that $M_e + \frac{Q}{c_i}\Lambda = M_{e+Q}$. More generally, $M_{e+\ell c_i+Q} = M_{e+\ell c_i} + \frac{Q}{c_i}\Lambda$ for all $0 \leq \ell < \frac{Q}{c_i}$. Combining this observation with the fact that $\{M_m\}_{m\geq 0}$ is periodic with respect to B, C, it follows that

$$\begin{array}{lcl} M_{e+\ell c_i+kQ} & = & M_{e+\ell c_i} + k\frac{Q}{c_i}\lambda, \text{ and} \\ M_{e+(\ell+1)c_i+kQ} & = & M_{e+(\ell+1)c_i} + k\frac{Q}{c_i}\lambda \end{array}$$

for all $k \geq 0$, $\ell \geq 0$. Consequently,

$$M_{e+(\ell+1)c_i+kQ} - M_{e+\ell c_i+kQ} \quad = \quad M_{e+(\ell+1)c_i} - M_{e+\ell c_i}$$

for all $k \geq 0$, $\ell \geq 0$. In other words, $M_{e+kc_i} = M_e + k\lambda$ for all $k \geq 0$. Combining it with $M_{b_i+kc_i} = M_{b_i} + k\lambda$ for all $0 \leq k \leq d$, where $b_i + dc_i = e$, we finally infer that $M_{b_i+kc_i} = M_{b_i} + k\lambda$ for all $k \geq 0$. This however implies that $L_i = \infty$, contradiction. Thus, $c_i \leq b_i \leq B + C + 3C^2 + 3BC$ in each iteration of the main loop. Notice that if $b_i \leq B + C$ and $b_{i+1} > B + C$, it takes at most C more iterations to return at line 12. Hence, the main loop is iterated at most $(B + C)^2 + C$ times in total.

Combining the bounds inferred in the above three cases, we obtain the bounds stated in this theorem. We finally prove if the algorithm terminates, the returned relation is indeed the transitive closure.

- R is not $*$-consistent, then here exists an integer $K > 0$ such that $R^K \Leftrightarrow \bot$. Then the algorithm will return at line 7, with the correct result:

$$P \vee \bigvee_{i=b+1}^{b+\ell c-1} R^i \Leftrightarrow \bigvee_{i=1}^{b} R^i \vee \bigvee_{i=b+1}^{b+\ell c-1} R^i \Leftrightarrow \bigvee_{i=1}^{b+\ell c-1} R^i$$

- R is $*$-consistent, then by previous arguments, the algorithm reaches the line 12 after at most $(B + C)^2 + C$ iterations of the main loop and returns the result:

$$\begin{array}{c} P \vee \exists k \geq 0 \,.\, \pi(k \cdot \Lambda + \sigma(R^b)) \circ \left(\bigvee_{j=0}^{c-1} R^j\right) \qquad \Leftrightarrow \\ \left(\bigvee_{i=1}^{b} R^i\right) \vee \left(\exists k \geq 0 \,.\, \pi(k \cdot \Lambda + \sigma(R^b)) \circ \left(\bigvee_{j=0}^{c-1} R^j\right)\right) \end{array}$$

  which is indeed the transitive closure of R, by Lemma 4 and Lemma 3. □

Algorithm 6 Closed Form of Periodic Relations

input a periodic relation R
output The closed form of R

1: **function** CLOSEDFORM(R)
2: **let** $P \leftarrow R$ **and** $b \leftarrow 1$ **and** $b_{jump} = 1$
3: **while** $true$ **do**
4: **for all** $c = 1, 2, \ldots, b$ **do**
5: **for all** $\ell = 0, 1, 2$ **do**
6: **if** $R^{b+\ell c} \Leftrightarrow \bot$ **then**
7: **return** $\widehat{R}(k, \mathbf{x}, \mathbf{x}') \Leftrightarrow \bigvee_{j=1}^{b+ic-1}(k = j) \wedge R^j$
8: **if** $\exists \Lambda \,.\, \sigma(R^b) + \Lambda = \sigma(R^{b+c}) \wedge \sigma(R^{b+c}) + \Lambda = \sigma(R^{b+2c})$ **then**
9: $K \leftarrow$ MAXCONSISTENT(R, b, Λ)
10: $L \leftarrow$ MAXPERIODIC(R, b, Λ, c, K)
11: **if** $L = \infty$ **then**
12: **return** $\widehat{R}(k, \mathbf{x}, \mathbf{x}') \Leftrightarrow$
$$\bigvee \left\{ \begin{array}{l} \bigvee_{i=1}^{b+c-1}(k = i) \wedge R^i \\ \exists \ell \geq 1 \,.\, \pi(\ell \cdot \Lambda + \sigma(R^b)) \circ \big(\bigvee_{j=0}^{c-1}(k = b + \ell c + j) \wedge R^j \big) \end{array} \right.$$
13: $b_{jump} \leftarrow \max\{b_{jump}, b + c \cdot (L + 1)\}$
14: $b_{next} \leftarrow \max\{b + 1, b_{jump}\}$
15: $P \leftarrow P \vee \bigvee_{i=b}^{b_{next}-1}(k = i \wedge R^i)$
16: $b \leftarrow b_{next}$
17: **function** MAXCONSISTENT(R,b,Λ)
18: **return** $\sup\{n \in \mathbb{N} \mid \rho(n \cdot \Lambda + \sigma(R^b)) \not\Leftrightarrow \bot\}$
19: **function** MAXPERIODIC(R, b, Λ, c, K)
20: **return** $\sup\{n \leq K \mid \forall 0 \leq \ell < n \,.\, \rho(\ell \cdot \Lambda + \sigma(R^b)) \circ R^c \Leftrightarrow \rho((\ell+1) \cdot \Lambda + \sigma(R^b))\}$

Algorithm 6 is a straightforward adaptation of Algorithm 5 that computes the closed form of a relation instead of its transitive closure, by modifying lines 7,12, and 15. Later, we use Algorithm 6 to compute transitive closures of finite monoid affine relations in Section 7.8.

6. Integer Relations

This section introduces three classes of integer relations, which are shown to be periodic next, in Section 7. We define first *difference bounds* relations, and generalize them to *octagonal relations*. The class of octagonal relations is non-deterministic, i.e. the next values are not functions of the current values of the variables. Finally, we introduce a deterministic class, namely *affine relations*, and concentrate on a semantic restriction of it, called the *finite monoid property*. All results in this section are proved elsewhere, and recalled here for the sake of self containment.

6.1. Difference Bounds Relations

Difference bounds constraints are known as *zones* in the context of timed automata verification [1] and abstract interpretation [43]. They are defined syntactically as conjunctions of atomic propositions of the form $x - y \leq c$, where x and y are variables and c is an integer constant. Difference bounds constraints can be represented as matrices and graphs. Moreover, their canonical form, useful for efficient inclusion checks, can be computed by the classical Floyd-Warshall algorithm. We report on these results in Section 6.1.1.

Difference bounds relations are defined as difference bounds constraints where variables can be also primed (e.g. $x - x' \leq 0$). The problem of computing transitive closures of difference bounds relations has been studied by Comon and Jurski [19] who showed that the transitive closure of a difference bounds relation is Presburger definable. Their proof was subsequently simplified in [15], using the notion of *zigzag* automata. Intuitively, zigzag automaton corresponding to a difference bounds relation R is a finite weighted automaton that encodes m-th power of R by minimal runs of length $m + 2$. Zigzag automata are also a reasoning tool use in Section **??** to prove periodicity of difference bounds relations. Furthermore, they also play a crucial role in deriving EXPTIME complexity upper bounds for Algorithm 5, for the class of difference bounds relations (Section 8).

We give the definitions of difference bounds relations and zigzag automata in Section 6.1.2 and Section 6.1.3, respectively. In the rest of this section, let $\mathbf{x} = \{x_1, x_2, ..., x_N\}$ be a set of variables ranging over $\mathbb{Z}$.

6.1.1. Difference Bounds Constraints

The following definition formalizes the notion of a difference bounds constraint.

Definition 8. *A formula $\phi(\mathbf{x})$ is a* difference bounds constraint *if it is equivalent to a finite conjunction of atomic propositions of the form $x_i - x_j \leq a_{ij},\ 1 \leq i, j \leq N, i \neq j$, where $a_{ij} \in \mathbb{Z}$.*

For instance, $x - y = 5$ is a difference bounds constraint, as it is equivalent to $x - y \leq 5 \ \wedge \ y - x \leq -5$. In practice, difference bounds constraints are represented either as matrices or as graphs:

Definition 9. *Let $\mathbf{x} = \{x_1, x_2, ..., x_N\}$ be a set of variables ranging over $\mathbb{Z}$ and $\phi(\mathbf{x})$ be a difference bounds constraint. Then a* difference bounds matrix *(DBM) representing ϕ is an $N \times N$ matrix M_ϕ such that:*

$$(M_\phi)_{i,j} = \begin{cases} \alpha_{i,j} & \text{if } (x_i - x_j \leq \alpha_{i,j}) \in AP(\phi) \\ \infty & \text{otherwise} \end{cases}$$

Definition 10. *Let $\mathbf{x} = \{x_1, x_2, ..., x_N\}$ be a set of variables ranging over $\mathbb{Z}$ and $\phi(\mathbf{x})$ be a difference bounds constraint. Then ϕ can be represented as a weighted graph $\mathcal{G}_\phi = (\mathbf{x}, \rightarrow)$, where each vertex corresponds to a variable, and there is an edge $x_i \xrightarrow{a_{ij}} x_j$ in $\mathcal{G}_\phi$ if and only if there exists a constraint $x_i - x_j \leq a_{ij}$ in ϕ. This graph is also called a* constraint graph.

Clearly, M_ϕ is the incidence matrix of $\mathcal{G}_\phi$. If $M \in \mathbb{Z}_\infty^{N\times N}$ is a DBM, the corresponding difference bounds constraint is defined as $\Phi_M \Leftrightarrow \bigwedge_{M_{ij}<\infty} x_i - x_j \leq M_{ij}$. We denote by $\|\phi\| = \sum_{(x_i - x_j \leq a_{ij}) \in AP(\phi)} |a_{ij}|$ the sum of absolute values of all coefficients of ϕ. The restriction of a DBM M_ϕ to variables $\mathbf{z} \subseteq \mathbf{x}$, denoted as $(M_\phi)_{\downarrow \mathbf{z}}$, is a matrix obtained by erasing the rows and columns of M_ϕ which encode constraints that involve variables $\mathbf{z} \setminus \mathbf{x}$. For two difference bounds matrices M_1, M_2, we write $M_1 = M_2$ if and only if $(M_1)_{ij} = (M_2)_{ij}$ for all $1 \leq i, j \leq N$ and $M_1 \leq M_2$ if and only if $(M_1)_{ij} \leq (M_2)_{ij}$ for all $1 \leq i, j \leq N$.

A DBM M is said to be *consistent* if and only if its corresponding constraint ϕ_M is consistent. The following proposition relates the consistency of ϕ to the existence of an elementary negative weight cycle of $\mathcal{G}_\phi$.

Proposition 2. *Let ϕ be a difference bounds constraint and $\mathcal{G}_\phi$ be the constraint graph of ϕ. Then, the following statements are equivalent:*

- *ϕ is consistent*
- *$\mathcal{G}_\phi$ contains an elementary negative weight cycle*

Proof: See e.g. [20], §25.5. □

The next definition gives a canonical form for consistent DBMs.

Definition 11. *A consistent DBM $M \in \mathbb{Z}_\infty^{N\times N}$ is said to be* closed *if and only if $M_{ii} = 0$ and $M_{ij} \leq M_{ik} + M_{kj}$, for all $1 \leq i, j, k \leq N$.*

Given a consistent DBM $M \in \mathbb{Z}^N \times \mathbb{Z}^N$, we denote the (unique) closed DBM by M^*. It is well-known that, if M is consistent, then M^* is unique, and can be computed from M in time $\mathcal{O}(N^3)$, by the classical Floyd-Warshall algorithm [20]. Consistency of M can be checked by the Floyd-Warshall algorithm too. By Proposition 2, it amounts to checking whether $M^*_{ii} < 0$ for some $1 \leq i \leq N$. The closed form is needed to check the equivalence and entailment of two difference bounds constraints.

Proposition 3 ([43]). *Let ϕ_1 and ϕ_2 be consistent difference bounds constraints. Then,*

- *$\phi_1 \Leftrightarrow \phi_2$ if and only if $M^*_{\phi_1} = M^*_{\phi_2}$,*
- *$\phi_1 \Rightarrow \phi_2$ if and only if $M^*_{\phi_1} \leq M^*_{\phi_2}$.*

The following proposition shows that given a difference bounds constraint $\phi(\mathbf{x})$, the formula $\exists x_k.\phi$ is a difference bounds constraint as well, and its closed DBM is effectively computable from M^*_ϕ.

Proposition 4. *Let $\phi(\mathbf{x})$, $\mathbf{x} = \{x_1, \ldots, x_N\}$, be a consistent difference bounds constraint. Further, let $1 \leq k \leq N$ and M' be the restriction of M^*_ϕ to $\mathbf{x} \setminus \{x_k\}$. Then, M' is closed and $\Phi(M') \Leftrightarrow \exists x_k.\phi(\mathbf{x})$. Moreover, the constraint graph $\mathcal{G}'$ corresponding to $\Phi(M')$ is obtained by erasing the vertex x_k together with the incident arcs from the graph $\mathcal{G}_{\Phi(M^*_\phi)}$.*

Proof: We use the following notation: for a $N \times N$ DBM M we denote by $\gamma(M) = \{\mathbf{v} \in \mathbb{Z}^N \mid v_i - v_j \leq m_{i,j}, 1 \leq i,j \leq N\}$ the set of *concretizations* of M. Notice that a DBM M is consistent if and only if $\gamma(M) \neq \emptyset$.

Clearly, M' is the incidence matrix of $\mathcal{G}'$. Without loss of generality, we assume that $k = N$. It is sufficient to show that:

$$\gamma(M') = \{(v_1, v_2, \ldots, v_{N-1}) \mid (v_1, v_2, \ldots, v_{N-1}, v) \in \gamma(M^*_\phi) \text{ for some } v \in \mathbb{Z}\}$$

The "$\supseteq$" direction is obvious, since M' is the restriction of M^*_ϕ to $\{x_1, x_2, \ldots, x_{N-1}\}$. For the "$\subseteq$" direction, we must show that there exists $v \in \mathbb{Z}$ such that $v_i - v \leq (M^*_\phi)_{i,N}$ and $v - v_j \leq (M^*_\phi)_{N,j}$, for all $1 \leq i,j \leq N$. But this amounts to $v_i - (M^*_\phi)_{i,N} \leq (M^*_\phi)_{N,j} + v_j$, for all $1 \leq i,j \leq N$. Since $(v_1, v_2, \ldots, v_{N-1}) \in \gamma(M')$ we have $v_i - v_j \leq (M^*_\phi)_{i,j}$, for all $1 \leq i,j \leq n$. Since M^*_ϕ is closed, $(M^*_\phi)_{i,j} \leq (M^*_\phi)_{i,N} + (M^*_\phi)_{N,j}$, which leads to the conclusion. DBM M' is closed as a direct consequence of the fact that M^*_ϕ is closed. □

6.1.2. Difference Bounds Relations and Their Powers

We first define difference bounds relations.

Definition 12. *Let $\mathbf{x} = \{x_1, \ldots, x_N\}$ be a set of variables. A relation $R \in \mathbb{Z}^N \times \mathbb{Z}^N$ is a* difference bounds relation *if it can be defined by a difference bounds constraint $R(\mathbf{x}, \mathbf{x}')$.*

The class of relations defined by difference bounds constraints over the variables $\mathbf{x} \cup \mathbf{x}'$ is denoted $\mathcal{R}_{db}$ in the following. A consequence of Proposition 4 is that $\mathcal{R}_{db}$ is closed under composition.

Proposition 5. *$\mathcal{R}_{db}$ is closed under intersection and composition.*

Proof: Let $R_1(\mathbf{x}, \mathbf{x}')$, $R_2(\mathbf{x}, \mathbf{x}')$ be difference bounds constraints defining difference bounds relations. By Definition 8, the conjunction $R_1(\mathbf{x}, \mathbf{x}') \wedge R_2(\mathbf{x}, \mathbf{x}')$ is a difference bounds constraint too. The composition of relations $R_1 \circ R_2$ can be defined as $\exists \mathbf{x}'' \,.\, (R_1(\mathbf{x}, \mathbf{x}'') \wedge R_2(\mathbf{x}'', \mathbf{x}'))$ which is again a difference bounds constraint, by Definition 8 and Proposition 4. □

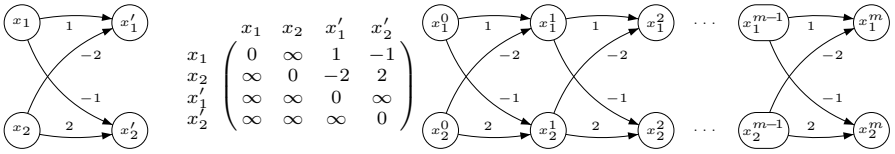
Example 4. *Let $R(x_1, x_2, x'_1, x'_2) \Leftrightarrow x_1 - x'_1 \leq 1 \wedge x_1 - x'_2 \leq -1 \wedge x_2 - x'_1 \leq -2 \wedge x_2 - x'_2 \leq 2$ be a difference bounds relation. Figure 4a shows the graph representation $\mathcal{G}_R$ and Figure 4b the closed DBM representation of R.* □

Given a difference bounds relation $R(\mathbf{x}, \mathbf{x}')$, we define the m-times concatenation of $\mathcal{G}_R$ with itself.

Definition 13. *Let $R(\mathbf{x}, \mathbf{x}')$, $x = \{x_1, \ldots, x_N\}$, be a difference bounds relation and $\mathcal{G}_R$ be its constraint graph. The m-*times unfolding *of $\mathcal{G}_R$ is defined as*

$$\mathcal{G}^m_R = (\bigcup_{k=0}^{N} \mathbf{x}^{(k)}, \rightarrow),$$

where $\mathbf{x}^{(k)} = \{x_i^{(k)} \mid 0 \leq i \leq N\}$ and for all $0 \leq k < N$,



(a) $\mathcal{G}_R$ (b) M_R^* (c) $\mathcal{G}_R^m$

Figure 4: Graph and matrix representation of a relation. Graph unfolding.

- $(x_i^{(k)} \xrightarrow{c} x_j^{(k)}) \in \rightarrow$ *if and only if* $(x_i - x_j \leq c) \in AP(\phi)$
- $(x_i^{(k)} \xrightarrow{c} x_j^{(k+1)}) \in \rightarrow$ *if and only if* $(x_i - x'_j \leq c) \in AP(\phi)$
- $(x_i^{(k+1)} \xrightarrow{c} x_j^{(k)}) \in \rightarrow$ *if and only if* $(x'_i - x_j \leq c) \in AP(\phi)$
- $(x_i^{(k+1)} \xrightarrow{c} x_j^{(k+1)}) \in \rightarrow$ *if and only if* $(x'_i - x'_j \leq c) \in AP(\phi)$

Each constraint in R^m corresponds to a path between extremal points in $\mathcal{G}_R^m$. Notice that, since $\mathcal{R}_{db}$ is closed under relational composition, then $R^m \in \mathcal{R}_{db}$ for any $m > 0$. Then we have:

$$R^m \Leftrightarrow \bigwedge_{1 \leq i,j \leq N} \begin{array}{r} x_i - x_j \leq \min\{x_i^0 \rightarrow x_j^0\} \ \wedge \ x'_i - x'_j \leq \min\{x_i^m \rightarrow x_j^m\} \ \wedge \\ x_i - x'_j \leq \min\{x_i^0 \rightarrow x_j^m\} \ \wedge \ x'_i - x_j \leq \min\{x_i^m \rightarrow x_j^0\} \end{array}$$

where $\min\{x_i^p \rightarrow x_j^q\}$ is the minimal weight between all paths among the extremal vertices x_i^p and x_j^q in $\mathcal{G}_R^m$, for $p, q \in \{0, m\}$.

Example 5. *Figure 4c depicts the m-times unfolding of $\mathcal{G}_R$ for the relation $R \Leftrightarrow x_1 - x'_1 \leq 1 \wedge x_1 - x'_2 \leq -1 \wedge x_2 - x'_1 \leq -2 \wedge x_2 - x'_2 \leq 2$.* □

The set of paths between any two extremal points in $\mathcal{G}_R^m$ can be seen as words over the finite alphabet of subgraphs of $\mathcal{G}_R^m$ that are accepted by a finite weighted automaton called *zigzag automaton* [15]. In the following section, we give the definition of these automata.

6.1.3. Zigzag Automata

This section defines zigzag automata, which can seen as recognizers of powers of difference bounds relations. Intuitively, a zigzag automaton corresponding to a difference bounds relation R is a finite weighted automaton that encodes m-th power of R by minimal runs of length $m + 2$.

6.1.4. Alphabet and Words

Without losing generality, in the following we work with a simplified (yet equivalent) form of difference bounds relations: all constraints of the form $x - y \leq \alpha$ are replaced by $x - t' \leq \alpha \ \wedge \ t' - y \leq 0$, and all constraints of the form $x' - y' \leq \alpha$ are replaced by $x' - t \leq \alpha \ \wedge \ t - y' \leq 0$, by introducing fresh variables $t \notin \mathbf{x}$. In other words, we can assume that the constraint graph $\mathcal{G}_R$ corresponding to R is *bipartite*, i.e. it does only contain edges from $\mathbf{x}$ to $\mathbf{x}'$ and vice versa.

A path π in $\mathcal{G}_R^m$ between, say, x^0 and y^m, with $x, y \in \mathbf{x}$ is represented by a word $w = w_1 \dots w_m$ of length m, as follows: the w_i symbol represents *simultaneously* all edges of π that involve only nodes from $\mathbf{x}^{i-1} \cup \mathbf{x}^i$, $1 \leq i \leq m$. Since we assumed that $\mathcal{G}_R^m$ is bipartite, it is easy to see that, for a path from x^0 to y^m, coded by a word w, the number of times the w_i symbol is traversed by the path is odd, whereas for a path from x^0 to y^0, or from x^m to y^m, this number is even. Hence the names of *even* and *odd automata*.

Given a difference bounds relation R, the *even alphabet* of R, denoted as Σ_R^e, is the set of all graphs satisfying the following conditions, for each $G \in \Sigma_R^e$:

1. the set of nodes of G is $\mathbf{x} \cup \mathbf{x}'$
2. for any $x, y \in \mathbf{x} \cup \mathbf{x}'$, there is an edge labeled with $\alpha \in \mathbb{Z}$ from x to y, only if the constraint $x - y \leq \alpha$ occurs in ϕ
3. the in-degree and out-degree of each node are at most one
4. the number of edges from $\mathbf{x}$ to $\mathbf{x}'$ equals the number of edges from $\mathbf{x}'$ to $\mathbf{x}$

Notice that the number of edges in all symbols of Σ_R^e is even.

The *odd alphabet* of R, denoted by Σ_R^o, is defined in the same way, with the exception of the last condition, which becomes:

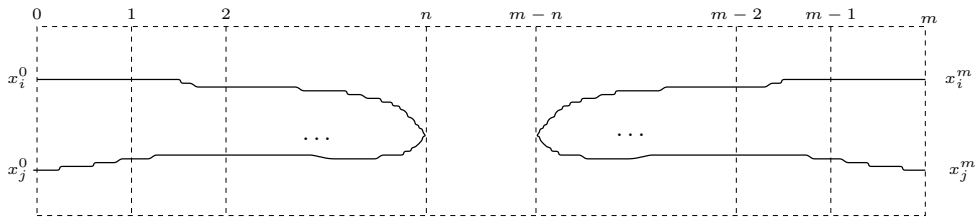
4. the difference between the number of edges from $\mathbf{x}$ to $\mathbf{x}'$ and the number of edges from $\mathbf{x}'$ to $\mathbf{x}$ is either 1 or -1

Notice that the number of edges in all symbols of Σ_R^o is odd.

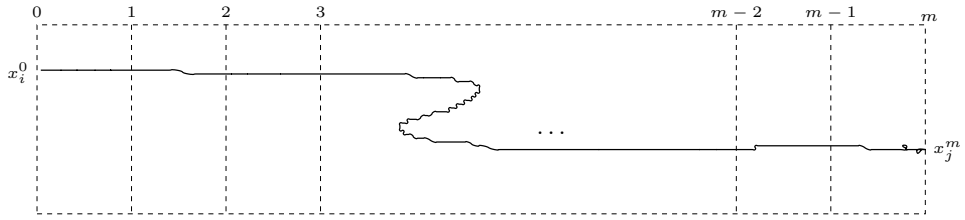
Let $\Sigma_R = \Sigma_R^e \cup \Sigma_R^o \cup \{\epsilon\}$ be the alphabet of the zigzag automaton for R, where ϵ is a special symbol of weight 0. The weight of any symbol $G \in \Sigma_R^e \cup \Sigma_R^o$, denoted $\omega(G)$, is the sum of the weights that occur on its edges. For a word $w = w_1 w_2 \dots w_n \in \Sigma_R^*$, we define its weight as $\omega(w) = \sum_{i=1}^n \omega(w_i)$.

6.1.5. Construction of Zigzag Automata

We are now ready for the definition of automata recognizing words that represent encodings of paths from $\mathcal{G}_R^m$. The *even automaton* recognizes paths that start and end on the same side of $\mathcal{G}_R^m$ i.e., either paths from x_i^0 to x_j^0, or from x_i^m to x_j^m, for some $1 \leq i, j \leq N$, respectively. We call the automata recognizing paths from x_i^0 to x_j^0 *forward* even automata, and the ones recognizing paths from x_i^m to x_j^m *backward* even automata (Figure 5 (a)). The *odd automata* recognize paths from one side of $\mathcal{G}_R^m$ to another. The automata recognizing paths from x_i^0 to x_j^m are called *forward* odd automata, whereas the ones recognizing paths from x_i^m to x_j^0 are called *backward* odd automata (Figure 5 (b)).



(a)



(b)

Figure 5: Runs of Even and Odd Automata

The even and odd automata share the same alphabet and transition table, while the differences are in the sets of initial and final states. The common transition table is defined as $T_R = \langle Q, \Delta, w \rangle$, where Q is the set of control states defined as:

$$\begin{array}{rcl} Q & = & Q_g \cup \bigcup_{1 \leq i,j \leq N} (Q_{ij}^{ef} \cup Q_{ij}^{eb} \cup Q_{ij}^{of} \cup Q_{ij}^{ob}) \text{ where} \\ Q_g & = & \{l, r, lr, rl, \perp\}^N \\ Q_{ij}^{ef} & = & \{I_{ij}^{ef}, F^{ef}\} \qquad Q_{ij}^{eb} = \{I^{eb}, F_{ij}^{eb}\} \\ Q_{ij}^{of} & = & \{I_i^{of}, F_j^{of}\} \qquad Q_{ij}^{ob} = \{I_i^{ob}, F_j^{ob}\} \end{array}$$

The $\{l, r, lr, rl, \perp\}$ components of states in Q_g capture the direction of incoming and outgoing edges (l for a path traversing from right to left, r for a path traversing from left to right, lr for a right incoming and right outgoing path, rl for a left incoming and left outgoing path, and $\perp$ when there are no incoming nor outgoing edges from that node.). Given $1 \leq i, j \leq N$, the sets $Q_{ij}^{ef}, Q_{ij}^{eb}, Q_{ij}^{of}, Q_{ij}^{ob}$ contain the initial and the final state in even forward (ef), even backward (eb), odd forward (of), and odd backward (ob) zigzag automaton corresponding to i, j, respectively. The four automata recognize paths from $x_i^{(0)}$ to $x_j^{(0)}$ (ef), from $x_i^{(0)}$ to $x_j^{(0)}$ (eb), from $x_i^{(0)}$ to $x_j^{(m)}$ (of), and from $x_i^{(m)}$ to $x_j^{(0)}$ (ob) in $\mathcal{G}_R^m$, respectively.

The set of transitions Δ is defined as:

$$\Delta = \Delta_g \cup \Delta_l \bigcup_{1 \leq i,j \leq N} (\Delta_{ij}^{ef} \cup \Delta_{ij}^{eb} \cup \Delta_{ij}^{of} \cup \Delta_{ij}^{ob})$$

There is a transition

$$\langle q_1 \ldots q_N \rangle \xrightarrow{G} \langle q'_1, \ldots, q'_N \rangle$$

in Δ_g if and only if the following conditions hold, for all $1 \leq i \leq N$:

- $q_i = l$ iff G has one edge whose destination is x_i, and no other edge involving x_i.
- $q'_i = l$ iff G has one edge whose source is x'_i, and no other edge involving x'_i.
- $q_i = r$ iff G has one edge whose source is x_i, and no other edge involving x_i.
- $q'_i = r$ iff G has one edge whose destination is x'_i, and no other edge involving x'_i.
- $q_i = lr$ iff G has exactly two edges involving x_i, one having x_i as source, and another as destination.
- $q'_i = rl$ iff G has exactly two edges involving x'_i, one having x'_i as source, and another as destination.
- $q'_i \in \{lr, \perp\}$ iff G has no edge involving x'_i.
- $q_i \in \{rl, \perp\}$ iff G has no edge involving x_i.

Some even paths in $\mathcal{G}_R^m$ may be of length strictly less than m. Since we want to recognize these path by runs of length $m+2$, we need several zero weight self-loop transitions:

$$\Delta_l = \{F^{ef} \xrightarrow{\epsilon} F^{ef}, I^{eb} \xrightarrow{\epsilon} I^{eb}\}$$

Finally, we define for each $q \leq i, j \leq N$ and each of the four zigzag automata (ef, eb, of, ob), the set of transitions that are incident with an initial or a final control state of the respective automaton:

$$\Delta_{ij}^{ef} = \begin{cases} \{I_{ij}^{ef} \xrightarrow{\epsilon} q \mid q_i = r, q_j = l, q_h \in \{lr, \perp\}, 1 \leq h \leq N, h \notin \{i, j\}\} & \text{if } i \neq j \\ \{I_{ij}^{ef} \xrightarrow{\epsilon} q \mid q_i = q_j = lr, q_h \in \{lr, \perp\}, 1 \leq h \leq N, h \neq i\} & \text{if } i = j \end{cases}$$

$$\bigcup \{q \xrightarrow{\epsilon} F^{ef} \mid q \in \{rl, \perp\}^N\}$$

$$\Delta_{ij}^{eb} = \begin{cases} \{q \xrightarrow{\epsilon} F_{ij}^{eb} \mid q_i = l, q_j = r, q_h \in \{lr, \perp\}, 1 \leq h \leq N, h \notin \{i, j\}\} & \text{if } i \neq j \\ \{q \xrightarrow{\epsilon} F_{ij}^{eb} \mid q_i = q_j = lr, q_h \in \{lr, \perp\}, 1 \leq h \leq N, h \neq i\} & \text{if } i = j \end{cases}$$

$$\bigcup \{I^{eb} \xrightarrow{\epsilon} q \mid q \in \{rl, \perp\}^N\}$$

$$\begin{aligned} \Delta_{ij}^{of} = \; & \{I_i^{of} \xrightarrow{\epsilon} q \mid q_i = r \text{ and } q_h \in \{lr, \perp\}, 1 \leq h \leq N, h \neq i\} \\ & \bigcup \{q \xrightarrow{\epsilon} F_j^{of} \mid q_j = r \text{ and } q_h \in \{rl, \perp\}, 1 \leq h \leq N, h \neq j\} \end{aligned}$$

$$\begin{aligned} \Delta_{ij}^{ob} = \; & \{I_i^{ob} \xrightarrow{\epsilon} q \mid q_i = l \text{ and } q_h \in \{lr, \perp\}, 1 \leq h \leq N, h \neq i\} \\ & \bigcup \{q \xrightarrow{\epsilon} F_j^{ob} \mid q_j = l \text{ and } q_h \in \{rl, \perp\}, 1 \leq h \leq N, h \neq j\} \end{aligned}$$

The weight function w maps each transition $q \xrightarrow{a} q' \in \Delta$, $q, q' \in Q, a \in \Sigma_R$ to $w(a)$.

Finally, for each $1 \leq i, j \leq N$, we define four zigzag automata

$$\begin{array}{lcllcl} A_{ij}^{ef} & = & \langle Q, \Delta, w, I_{i,j}^{ef}, F^{ef} \rangle & A_{ij}^{of} & = & \langle Q, \Delta, w, I_i^{of}, F_j^{of} \rangle \\ A_{ij}^{eb} & = & \langle Q, \Delta, w, I^{eb}, F_{i,j}^{eb} \rangle & A_{ij}^{ob} & = & \langle Q, \Delta, w, I_i^{ob}, F_j^{ob} \rangle \end{array}$$

Notice that these automata share the same states and transitions, and the number of states is at most $5^N + 2N^2 + 4N + 2$, where N is the number of variables in $\mathbf{x}$.

6.1.6. Language of Zigzag Automata

Recall that $\mathcal{G}_R^m$ denotes the constraint graph corresponding to R^m, obtained by concatenating the constraint graph of R to itself $m > 0$ times. We say that a path in $\mathcal{G}_R^m$ *stretches between* k *and* l, for some $k \leq l$, if the path contains at least one node from $\mathbf{x}^i$, for each $k \leq i \leq l$ and contains no node from $\mathbf{x}^i$, for each i such that $i < k$ or $i > l$. Intuitively, all paths from x_i^0 to x_j^0 in $\mathcal{G}_R^m$ are recognized by the automaton A_{ij}^{ef}, paths from x_i^m to x_j^m by A_{ij}^{eb} (Figure 5 (a)), paths from x_i^0 to x_j^m by A_{ij}^{of}, and paths from x_i^m to x_j^0 by A_{ij}^{ob} (Figure 5 (b)). The following lemma makes the relationship between between paths in $\mathcal{G}_R^m$ and runs in zigzag automata of length $m + 2$ precise.

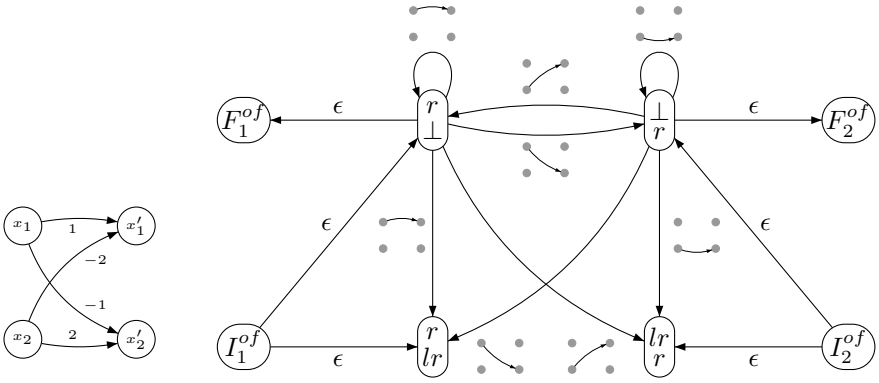
Lemma 5 ([15]). *Suppose that $\mathcal{G}_R^m$ does not have cycles of negative weight, for some $m > 0$. Then, for any $1 \leq i, j \leq N$, $i \neq j$, the following hold:*

1. *A_{ij}^{ef} has an accepting run of length $m+2$ if and only if there exists a path in $\mathcal{G}_R^m$, from x_i^0 to x_j^0, that stretches between 0 and n, for some $0 \leq n \leq m$. Moreover, the minimal weight among all paths from x_i^0 to x_j^0 in $\mathcal{G}_R^m$, stretching from 0 to n, for some $0 \leq n \leq m$, equals the minimal weight among all accepting runs of A_{ij}^{ef} of length $m + 2$.*
2. *A_{ij}^{eb} has an accepting run of length $m+2$ if and only if there exists a path in $\mathcal{G}_R^m$, from x_i^m to x_j^m, that stretches between n and m, for some $0 \leq n \leq m$. Moreover, the minimal weight among all paths from x_i^m to x_j^m in $\mathcal{G}_R^m$, stretching from n to m, for some $0 \leq n \leq m$, equals the minimal weight among all accepting runs of A_{ij}^{eb}, of length $m + 2$.*
3. *A_{ij}^{of} has an accepting run of length $m + 2$ if and only if there exists a path in $\mathcal{G}_R^m$, from x_i^0 to x_j^m. Moreover, the minimal weight among all paths from x_i^0 to x_j^m in $\mathcal{G}_R^m$ equals the minimal weight among all accepting runs of length $m + 2$.*
4. *A_{ij}^{ob} has an accepting run of length $m+2$ if and only if there exists a path in $\mathcal{G}_R^m$, from x_i^m to x_j^0. Moreover, the minimal weight among all paths from x_i^m to x_j^0 in $\mathcal{G}_R^m$ equals the minimal weight among all accepting runs of length $m + 2$.*

Proof: See [15], Lemmas 4.3, 4.4, 4.6 and 4.7. □

Example 6. *Let us show the construction of the zigzag automaton for the relation $R \Leftrightarrow x_1 - x_1' \leq 1 \wedge x_1 - x_2' \leq -1 \wedge x_2 - x_1' \leq -2 \wedge x_2 - x_2' \leq 2$. Figures 5(a) and (b) depict $\mathcal{G}_R$ and M_R^*. Notice that there are only forward odd paths, i.e. paths from*

$\mathbf{x_0}$ *to* $\mathbf{x}_m$ *in* $\mathcal{G}_R^m$ *for any* $m \geq 1$. *The transition table* $T_R = \langle Q, \Delta, w \rangle$ *of the zigzag automaton is depicted in Figure 6 (isolated states, such as* (r, l)*, have been removed). For instance, the automaton* $A_{xy}^{ef} = \langle T_R, I_x^{of}, F_x^{of} \rangle$ *recognizes a run of length* $m+2$ *with weight* w *if and only if there is a path from* x_0 *to* x^m *in* $\mathcal{G}_R^m$ *of length* m *and with weight* w. *There are four such paths in* $\mathcal{G}_R^3$ *and the Figure 7 shows the corresponding runs of the zigzag automaton. The second and the third runs have minimal weight.* □



(a) $\mathcal{G}_R$ (b) T_R

Figure 6: Zigzag automaton

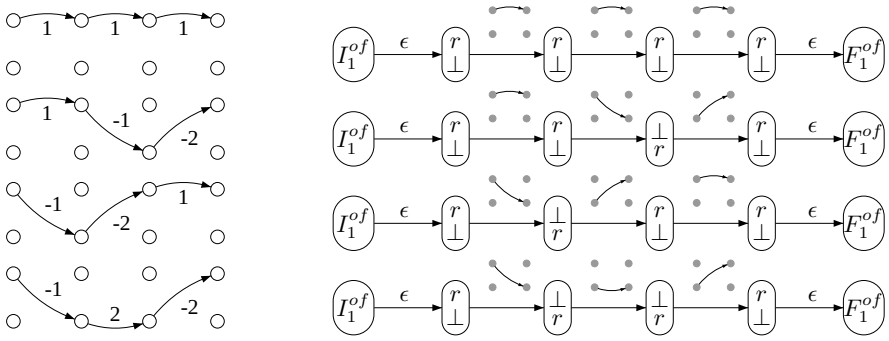


Figure 7: Runs

6.2. Octagonal Relations

Octagonal constraints (also known as Unit Two Variables Per Inequality or UTVPI, for short) appear in the context of abstract interpretation where they have been extensively studied as an abstract domain [43]. They are defined syntactically as a conjunctions of atomic propositions of the form $\pm x \pm y \leq c$, where x and y are variables and c is an integer constant. Thus, they can be seen as a generalization of difference bounds

constraints. We adopt the classical representation of octagonal constraints (or octagons, for short) $\phi(x_1, \ldots, x_N)$ as difference bounds constraints $\phi(y_1, \ldots, y_{2N})$, where y_{2i-1} stands for $+x_i$ and y_{2i} stands for $-x_i$ with an implicit condition $y_{2i-1} = -y_{2i}$, for each $1 \leq i \leq N$. With this convention, [5] provides an algorithm for computing the canonical form of an octagon, by first computing the canonical form of the corresponding difference bounds constraint and subsequently *tightening* the difference bounds constraints $y_i - y_j \leq c$. We present these results in Section 6.2.1.

Octagonal relations are defined as octagonal constraints where variables can be also primed. Octagonal relations were studied in [11] where it was shown that the transitive closure is Presburger definable. The core result of [11] is that the canonical form of the m-th power of an octagonal relation R can be computed directly from the m-th power of a difference bounds relation that represents R. We present these results in Section 6.2.2.

6.2.1. Octagonal Constraints

Let $\mathbf{x} = \{x_1, x_2, ..., x_N\}$ be a set of variables ranging over $\mathbb{Z}$. The class of integer octagonal constraints is defined as follows:

Definition 14. *A formula $\phi(\mathbf{x})$ is an* octagonal constraint *if it is equivalent to a finite conjunction of terms of the form $x_i - x_j \leq a_{ij}$, $x_i + x_j \leq b_{ij}$ or $-x_i - x_j \leq c_{ij}$ where $a_{ij}, b_{ij}, c_{ij} \in \mathbb{Z}$, for all $1 \leq i, j \leq N$.*

We represent octagons as difference bounds constraints over the dual set of variables $\mathbf{y} = \{y_1, y_2, \ldots, y_{2N}\}$, with the convention that y_{2i-1} stands for x_i and y_{2i} for $-x_i$, respectively. For example, the octagonal constraint $x_1 + x_2 = 3$ is represented as $y_1 - y_4 \leq 3 \wedge y_2 - y_3 \leq -3$. In order to handle the $\mathbf{y}$ variables in the following, we define $\overline{i} = i - 1$, if i is even, and $\overline{i} = i + 1$ if i is odd. Obviously, we have $\overline{\overline{i}} = i$, for all $i \in \mathbb{Z},\ i \geq 0$. We denote by $\overline{\phi}(\mathbf{y})$ the difference bounds constraint over $\mathbf{y}$ that represents $\phi(\mathbf{x})$ and which is defined as follows:

Definition 15. *Given an octagonal constraint $\phi(\mathbf{x})$, $\mathbf{x} = \{x_1, \ldots, x_N\}$, its difference bounds representation $\overline{\phi}(\mathbf{y})$, $\mathbf{y} = \{y_1, \ldots, y_{2N}\}$ is a conjunction of the following difference bounds constraints where $1 \leq i \neq j \leq N$, $c \in \mathbb{Z}$.*

$$
\begin{array}{lcl}
(x_i - x_j \leq c) \in AP(\phi) & \Leftrightarrow & (y_{2i-1} - y_{2j-1} \leq c), (y_{2j} - y_{2i} \leq c) \in AP(\overline{\phi}) \\
(-x_i + x_j \leq c) \in AP(\phi) & \Leftrightarrow & (y_{2j-1} - y_{2i-1} \leq c), (y_{2i} - y_{2j} \leq c) \in AP(\overline{\phi}) \\
(-x_i - x_j \leq c) \in AP(\phi) & \Leftrightarrow & (y_{2i} - y_{2j-1} \leq c), (y_{2j} - y_{2i-1} \leq c) \in AP(\overline{\phi}) \\
(x_i + x_j \leq c) \in AP(\phi) & \Leftrightarrow & (y_{2i-1} - y_{2j} \leq c), (y_{2j-1} - y_{2i} \leq c) \in AP(\overline{\phi}) \\
(2x_i \leq c) \in AP(\phi) & \Leftrightarrow & (y_{2i-1} - y_{2i} \leq c) \in AP(\overline{\phi}) \\
(-2x_i \leq c) \in AP(\phi) & \Leftrightarrow & (y_{2i} - y_{2i-1} \leq c) \in AP(\overline{\phi})
\end{array}
$$

The following equivalence relates ϕ and $\overline{\phi}$:

$$
\phi(\mathbf{x}) \Leftrightarrow (\exists y_2, y_4, \ldots, y_{2N} \,.\, \overline{\phi} \wedge \bigwedge_{i=1}^{N} y_{2i-1} = -y_{2i})[x_i / y_{2i-1}]_{i=1}^{N} \qquad (2)
$$

An octagonal constraint ϕ is equivalently represented by the DBM $M_{\overline{\phi}} \in \mathbb{Z}_{\infty}^{2N \times 2N}$, corresponding to $\overline{\phi}$. We sometimes write M_{ϕ} instead of $M_{\overline{\phi}}$. We say that a DBM $M \in \mathbb{Z}_{\infty}^{2N \times 2N}$ is *coherent* iff $M_{ij} = M_{\overline{j}\overline{i}}$ for all $1 \leq i, j \leq 2N$. This property is needed since e.g. an atomic proposition $x_i - x_j \leq a_{ij}$, $1 \leq i, j \leq N$, can be represented as both $y_{2i-1} - y_{2j-1} \leq a_{ij}$ and $y_{2j} - y_{2i} \leq a_{ij}$. Dually, a coherent DBM $M \in \mathbb{Z}_{\infty}^{2N \times 2N}$ corresponds to the octagonal constraint:

$$\Omega_M \Leftrightarrow \bigwedge_{1 \leq i,j \leq N} (x_i - x_j \leq M_{2i-1,2j-1} \wedge x_i + x_j \leq M_{2i-1,2j} \wedge -x_i - x_j \leq M_{2i,2j-1}) \tag{3}$$

A coherent DBM M is said to be *octagonal-consistent* if and only if Ω_M is consistent. Similar to the case of difference bounds constraints, for an octagonal constraint ϕ, we define $\|\phi\|$ as $\|\phi\| \stackrel{def}{=} \|\overline{\phi}\|$, where $\|\overline{\phi}\|$ is the maximal absolute value of all coefficients of $\overline{\phi}$ defined in Section 6.1.

Definition 16. *An octagonal-consistent coherent DBM $M \in \mathbb{Z}_{\infty}^{2N \times 2N}$ is said to be* tightly closed *if and only if the following hold, for all $1 \leq i, j, k \leq 2N$:*

1. $M_{ii} = 0$
2. $M_{i\bar{i}}$ *is even*
3. $M_{ij} \leq M_{ik} + M_{kj}$
4. $M_{ij} \leq \lfloor \frac{M_{i\bar{i}}}{2} \rfloor + \lfloor \frac{M_{\bar{j}j}}{2} \rfloor$

Given an octagonal-consistent coherent DBM $M \in \mathbb{Z}^{2N} \times \mathbb{Z}^{2N}$, we denote the (unique) tightly closed DBM by M^t . The following theorem from [5] provides an effective way of testing octagonal-consistency and computing the tight closure of a coherent DBM. Moreover, it shows that the tight closure of a given DBM is unique and can also be computed in time $\mathcal{O}(N^3)$.

Theorem 3. *[5] Let $M \in \mathbb{Z}_{\infty}^{2N \times 2N}$ be a coherent DBM. Then M is octagonal-consistent if and only if M is consistent and $\lfloor \frac{M^*_{i\bar{i}}}{2} \rfloor + \lfloor \frac{M^*_{\bar{i}i}}{2} \rfloor \geq 0$, for all $1 \leq i \leq 2N$. Moreover, if M is octagonal-consistent, the tight closure of M is the DBM $M^t \in \mathbb{Z}_{\infty}^{2N \times 2N}$ defined as:*

$$M^t_{ij} = \min \left\{ M^*_{ij}, \left\lfloor \frac{M^*_{i\bar{i}}}{2} \right\rfloor + \left\lfloor \frac{M^*_{\bar{j}j}}{2} \right\rfloor \right\}$$

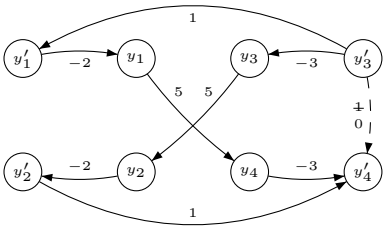
for all $1 \leq i, j \leq 2N$ where $M^* \in \mathbb{Z}_{\infty}^{2N \times 2N}$ is the closure of M.

The tight closure of DBMs is needed for checking equivalence and entailment between octagonal constraints.

Proposition 6 ([43]). *Let ϕ_1 and ϕ_2 be octagonal-consistent octagonal constraints. Then,*

- $\phi_1 \Leftrightarrow \phi_2$ *if and only if* $M^t_{\phi_1} = M^t_{\phi_2}$,
- $\phi_1 \Rightarrow \phi_2$ *if and only if* $M^t_{\phi_1} \leq M^t_{\phi_2}$.

It has been shown in [11] that octagonal constraints are closed under existential quantification.



$\mathcal{G}_R$

$$M_R^t = \begin{array}{c} \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{array} \begin{array}{c} \begin{array}{cccccccc} y_1 & y_2 & y_3 & y_4 & y'_1 & y'_2 & y'_3 & y'_4 \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 5 & \infty & \infty & \infty & 2 \\ \infty & 0 & \infty & \infty & \infty & -2 & \infty & -1 \\ \infty & 5 & 0 & \infty & \infty & 3 & \infty & 4 \\ \infty & \infty & \infty & 0 & \infty & \infty & \infty & -3 \\ -2 & \infty & \infty & 3 & 0 & \infty & \infty & 0 \\ \infty & \infty & \infty & \infty & \infty & 0 & \infty & 1 \\ -1 & 2 & -3 & 4 & 1 & 0 & 0 & 0 \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix} \end{array}$$

Figure 8: Graph and matrix representation of a relation.

Proposition 7. *Let $\phi(\mathbf{x})$, $\mathbf{x} = \{x_1, \ldots, x_N\}$, be an octagonal-consistent octagonal constraint. Further, let $1 \leq k \leq 2N$ and M' be the restriction of M^t_ϕ to $\mathbf{y} \setminus \{y_{2k-1}, y_{2k}\}$. Then, M' is tightly closed, and $\Omega(M') \Leftrightarrow \exists x_k.\phi(\mathbf{x})$.*

Proof: See [11], Theorem 2. □

6.2.2. Octagonal Relations and Their Powers

Definition 17. *Let $\mathbf{x} = \{x_1, \ldots, x_N\}$ be a set of variables. A relation $R \in \mathbb{Z}^N \times \mathbb{Z}^N$ is an* octagonal relation *if it can be defined by an octagonal constraint $R(\mathbf{x}, \mathbf{x}')$.*

The class of relations defined by octagonal constraints is denoted by $\mathcal{R}_{oct}$ in the following.

Example 7. *Consider the octagonal relation $R(x_1, x_2, x'_1, x'_2) \Leftrightarrow x_1 + x_2 \leq 5 \wedge x'_1 - x_1 \leq -2 \wedge x'_2 - x_2 \leq -3 \wedge x'_2 - x'_1 \leq 1$. Its difference bounds representation is $\overline{R}(\mathbf{y}, \mathbf{y}') \Leftrightarrow y_1 - y_4 \leq 5 \wedge y_3 - y_2 \leq 5 \wedge y'_1 - y_1 \leq -2 \wedge y_2 - y'_2 \leq -2 \wedge y'_3 - y_3 \leq -3 \wedge y_4 - y'_4 \leq -3 \wedge y'_3 - y'_1 \leq 1 \wedge y'_2 - y'_4 \leq 1$, where $\mathbf{y} = \{y_1, \ldots, y_4\}$. Figure 8a shows the graph representation $\mathcal{G}_R$. Note that the implicit constraint $y'_3 - y'_4 \leq 1$ (represented by a dashed edge in Figure 8a) is not tight. The tightening step replaces the bound 1 (crossed in Figure 8a) with 0. Figure 8b shows the tightly closed DBM representation of R, denoted M^t_R.* □

A consequence of Proposition 7 is that $\mathcal{R}_{oct}$ is closed under composition.

Proposition 8. *$\mathcal{R}_{oct}$ is closed under intersection and composition.*

Proof: Let $R_1(\mathbf{x}, \mathbf{x}')$, $R_2(\mathbf{x}, \mathbf{x}')$ be octagonal constraints defining octagonal relations. By Definition 14, $R_1(\mathbf{x}, \mathbf{x}') \wedge R_2(\mathbf{x}, \mathbf{x}')$ is an octagonal constraints to. The composition of relations $R_1 \circ R_2$ can be defined as $\exists \mathbf{x}'' \,.\, (R_1(\mathbf{x}, \mathbf{x}'') \wedge R_2(\mathbf{x}'', \mathbf{x}'))$ which is again an octagonal constraint, by Definition 14 and Proposition 7. □

We rely in the following chapters on the main result of [11], which establishes the following relation between $M^t_{\overline{R}^m}$ (the tightly closed octagonal DBM corresponding to the m-th iteration of R) and $M^*_{\overline{R}^m}$ (the closed DBM corresponding to the m-th iteration of the difference bounds relation $\overline{R}$), for all $m \geq 0$:

Theorem 4. *[11] Let $R(\mathbf{x}, \mathbf{x}')$, $\mathbf{x} = \{x_1, \ldots, x_N\}$, be a $*$-consistent octagonal relation. Then, $M^t_{\overline{R^m}} = M^t_{\overline{R}^m}$ for all $m \geq 0$. Consequently,*

$$(M^t_{\overline{R^m}})_{ij} = \min\left\{(M^*_{\overline{R}^m})_{ij}, \left\lfloor \frac{(M^*_{\overline{R}^m})_{i\bar{i}}}{2} \right\rfloor + \left\lfloor \frac{(M^*_{\overline{R}^m})_{\bar{j}j}}{2} \right\rfloor\right\}$$

for all $1 \leq i, j \leq 4N$.

This relation is in fact a generalization of the tight closure definition from Theorem 3, from $m = 1$ to any $m \geq 0$.

6.3. Finite Monoid Affine Relations

Sections 6.1 and 6.2 presented two classes of non-deterministic relations. In this section, we present linear affine relations which are a general model of deterministic transition relations. Linear affine relations are relations of the form $\mathbf{x}' = A \times \mathbf{x} + \mathbf{b} \wedge \phi(\mathbf{x})$, where $\mathbf{x}' = A \times \mathbf{x} + \mathbf{b}$ is an affine transformation and $\phi(\mathbf{x})$ is a Presburger guard. We present two subclasses of linear affine relations, called *finite monoid affine relations* and *polynomially bounded affine relations*.

The class of finite monoid affine relations was the first class of integer relations for which the transitive closure has been shown to be Presburger definable by Boigelot [9]. Informally, an affine relation is a finite monoid relation if the set of powers of its transformation matrix is finite. Originally, Boigelot characterized this class by two decidable conditions in [9] (we report on these conditions in Lemma 5). Later, Finkel and Leroux noticed in [25] that Boigelot's conditions correspond to the finite monoid property, which is also known to be decidable [42].

The second subclass of polynomially bounded relations is defined by dropping one of the Boigelot's conditions and by requiring that the guard of a relation is linear. We study this subclass in Chapter **??** which presents a method for computation of termination preconditions for this class.

Definition 18. *Let $\mathbf{x} = \langle x_1, \ldots, x_N \rangle$ be a vector of variables ranging over $\mathbb{Z}$. A relation $R \in \mathbb{Z}^N \times \mathbb{Z}^N$ is an* affine relation *if it can be defined by a formula $R(\mathbf{x}, \mathbf{x}')$ of the form*

$$R(\mathbf{x}, \mathbf{x}') \;\Leftrightarrow\; \mathbf{x}' = A \times \mathbf{x} + \mathbf{b} \;\wedge\; \phi(\mathbf{x}) \tag{4}$$

where $A \in \mathbb{Z}^{N \times N}$, $\mathbf{b} \in \mathbb{Z}^N$, and ϕ is a Presburger formula over unprimed variables only, called the guard*. The formula $\mathbf{x}' = A \times \mathbf{x} + \mathbf{b}$, defining a linear transformation, is called the* update*.*

The affine transformation is said to have the *finite monoid property* [9, 25] if the monoid of powers of A, denoted as $\langle \mathcal{M}_A, \times \rangle$, where $\mathcal{M}_A = \{A^i \mid i \geq 0\}$, is finite. In this case, we also say that A has the finite monoid property. Here $A^0 = I_N$ and $A^i = A \times A^{i-1}$, for $i > 0$. Intuitively, the finite monoid property is equivalent to the fact that A has finitely many powers (considering the standard integer multiplication). A linear affine relation has the finite monoid property if and only if the matrix A defining the update has the finite monoid property.

It has been shown in [25] that finite monoid property can be equivalently characterized by a pair of conditions. Before presenting this characterization, we recall several notions of linear algebra.

If $A \in \mathbb{Z}^{n\times n}$ is a square matrix, and $\mathbf{v} \in \mathbb{Z}^n$ is a column vector of integer constants, then any complex number $\lambda \in \mathbb{C}$ such that $A\mathbf{v} = \lambda\mathbf{v}$, for some complex vector $\mathbf{v} \in \mathbb{C}^n$, is called an *eigenvalue* of A. The vector $\mathbf{v}$ in this case is called an *eigenvector* of A. It is known that the eigenvalues of A are the roots of the *characteristic polynomial* $P_A(x) = \det(A - xI_n) = 0$, which is an effectively computable univariate polynomial. The *minimal polynomial* of A is the polynomial μ_A of lowest degree such that $\mu_A(A) = 0$. By the Cayley-Hamilton Theorem, the minimal polynomial always divides the characteristic polynomial, i.e. the roots of the former are root of the latter.

If $\lambda_1, \ldots, \lambda_m$ are the eigenvalues of A, then $\lambda_1^p, \ldots, \lambda_m^p$ are the eigenvalues of A^p, for all integers $p > 0$. A matrix is said to be *diagonalizable* if and only if there exists a non-singular matrix $U \in \mathbb{C}^{N\times N}$ and a diagonal matrix with the eigenvalues $\lambda_1, \ldots, \lambda_m$ occurring on the main diagonal, such that $A = U \times D \times U^{-1}$. This is the case if and only if μ_A has only roots of multiplicity one (see e.g. Thm 8.47 in [9]).

A complex number r is said to be a *root of the unity* if $r^d = 1$ for some integer $d > 0$. The *cyclotomic polynomial* $F_d(x)$ is the product of all monomials $(x - \omega)$, where $\omega^d = 1$, and $\omega^e \neq 1$, for all $0 < e < d$. It is known that a polynomial has only roots which are roots of unity if and only if it is a product of cyclotomic polynomials.

With these notions, the finite monoid property is defined by the following equivalent conditions.

Theorem 5. *[Thm 8.42 and 8.44 in [9] and Prop 2 in [25]] a relation $R \equiv A \times \mathbf{x} + \mathbf{b}$, where $A \in \mathbb{Z}^{N\times N}$ and $\mathbf{b} \in \mathbb{Z}^N$ has the finite monoid condition if and only if there exists $p > 0$ such that the following hold:*

1. *every eigenvalue of A^p belongs to the set $\{0, 1\}$,*
2. *the minimal polynomial $\mu_{A^p}(x)$ of A^p belongs to the set $\{0, x, x-1, x(x-1)\}$ (or, equivalently, A^p is diagonalizable).*

Both conditions in Theorem are decidable [9, 42].

In Chapter **??**, we study another subclass of affine relations with linear guards and transformation matrix whose eigenvalues are either zero or roots of the unity.

Definition 19. *If $\mathbf{x} = \langle x_1, \ldots, x_N\rangle$ is a vector of variables ranging over $\mathbb{Z}$, a* polynomially bounded affine relation *is a relation of the form*

$$R(\mathbf{x}, \mathbf{x}') \;\Leftrightarrow\; \mathbf{x}' = A \times \mathbf{x} + \mathbf{b} \;\wedge\; C\mathbf{x} \geq \mathbf{d} \tag{5}$$

where $A \in \mathbb{Z}^{n\times n}$, $C \in \mathbb{Z}^{p\times n}$ are matrices, and $\mathbf{b} \in \mathbb{Z}^n$, $\mathbf{d} \in \mathbb{Z}^p$ are column vectors of integer constants and moreover, all eigenvalues of A are either zero or roots of the unity.

Note that if A is a finite monoid matrix, then all eigenvalues of A are either zero or roots of the unity. Thus, the condition on A is weaker for polynomially bounded affine relations. However, since the guard of finite monoid relations is more general (Presburger), the two classes are incomparable.

7. Periodicity of Integer Relations

This section is dedicated to instantiations of Algorithm 5 from Chapter **??** to three classes of arithmetic relations for which the transitive closure is known to be definable in Presburger arithmetic: difference bounds relations, octagonal relations, and finite monoid affine transformations. To compute the transitive closure of these relations using Algorithm 5, one first needs to prove that the three classes are periodic, otherwise termination of Algorithm 5 is not guaranteed. Our proofs rely mostly on a fact that any matrix is periodic when its powers are computed in the tropical semiring $(\mathbb{Z}_\infty, \min, +, \infty, 0)$. The intuition behind periodicity of difference bounds relations is that the k-th power of a relation from this class can be encoded by minimal runs of length k in *zigzag automata* which in turn can be computed as the k-th tropical power of the incidence matrix of the automaton. Thus, periodicity of the sequence of tropical powers of the incidence matrix entails periodicity of a difference bounds relation. Periodicity of the three classes thus provides common grounds to the acceleration problem and also gives shorter proofs for the fact that the transitive closures of these three classes are definable in Presburger arithmetic.

The efficiency of Algorithm 5 depends on two factors. Given a relation with prefix b and period c, Theorem 2 proves that Algorithm 5 makes $O((b+c)^2)$ iterations of the main loop. Thus, the prefix b and the period c are important complexity parameters, and we give asymptotic bounds for them in Chapter **??**. For difference bounds and octagonal relations, these bounds are closely related to bounds on the prefix and the period of the incidence matrix of zigzag automata. This chapter therefore gives an alternative proof to the fact that each matrix is periodic in the tropical semiring which moreover gives asymptotic bounds.

Another important efficiency factor is the complexity of the procedures MAXCONSISTENT and MAXPERIODIC, which are called by Algorithm 5 to detect the maximal interval that is periodic with respect to the current prefix and period candidates. In general, for all three classes of relations we consider, these procedures can be implemented using Presburger arithmetic queries. However, in practice, one would like to avoid as much as possible using Presburger solvers, due to reasons of high complexity of decision procedures for Presburger arithmetic. In this chapter, we give direct decision methods which avoid calls to external Presburger or SMT solvers completely and which are of polynomial time complexity in the size of the prefix, period, $\|R\|$, and N, where $\|R\|$ denotes the sum of absolute values of the coefficients of a relation R and N denotes the number of variables used to define a given relation R.

Roadmap.. In Section 7.1, we prove that every matrix is periodic in the tropical semiring and establish asymptotic bounds on the size of its prefix and period. Next, Sections 7.2, 7.5, and 7.8 study the classes of difference bounds, octagonal, and finite monoid affine relations, respectively. In each of these sections, we prove that the respective class is periodic, present implementations of the MAXCONSISTENT and MAXPERIODIC procedures, and study their complexity. We defer all experiments with Algorithm 5 to Chapter **??**.

7.1. Periodicity of Matrices

In this section, we prove that each matrix is periodic when its powers are computed in the *tropical semiring* which is defined as follows.

An *idempotent semiring* is a set $(\mathcal{S}, +, \cdot, \mathbf{0}, \mathbf{1})$ equipped with two operations, the addition $+$ and the multiplication $\cdot$, such that $(\mathcal{S}, +, \mathbf{0})$ is an idempotent (i.e., $p+p=p$ for all $p \in \mathcal{S}$) commutative monoid with neutral element $\mathbf{0}$ and $(\mathcal{S}, \cdot, \mathbf{1})$ is a monoid with neutral element $\mathbf{1}$. Moreover, multiplication distributes both left and right over addition and $\mathbf{0} \cdot r = r \cdot \mathbf{0} = \mathbf{0}$, for all $r \in \mathcal{S}$. The *tropical semiring*[3] is an idempotent semiring $(\mathbb{Z}_\infty, \min, +, \infty, 0)$ [52] with the extended arithmetic operations $x+\infty = \infty$, and $\min(x, \infty) = x$, for all $x \in \mathbb{Z}$, where $\min(x, y)$ denotes the minimum between the values x and y.

If S is a set, let $S^{m\times m}$ denote the set of square matrices of size m, with entries in S. For two matrices $A, B \in \mathbb{Z}_\infty^{m\times m}$, we define the sum $(A+B)_{ij} = A_{ij} + B_{ij}$. The classical product is defined for $A, B \in \mathbb{Z}^{m\times m}$ as $(A \times B)_{ij} = \sum_{k=1}^{m}(a_{ik} \cdot b_{kj})$. The *tropical product* is defined for $A, B \in \mathbb{Z}_\infty^{m\times m}$ as $(A \boxtimes B)_{ij} = \min_{k=1}^{m}(a_{ik} + b_{kj})$. Let $\mathbf{I}_m \in \mathbb{Z}^{m\times m}$ be the identity matrix, i.e. $\mathbf{I}_{ii} = 1$ and $\mathbf{I}_{ij} = 0$, for all $1 \leq i, j \leq m$, $i \neq j$, and $\mathbb{I}_m \in \mathbb{Z}_\infty^{m\times m}$ be the tropical identity matrix, i.e. $\mathbb{I}_{ii} = 0$ and $\mathbb{I}_{ij} = \infty$, for all $1 \leq i, j \leq m$, $i \neq j$. Then we define $A^0 = \mathbf{I}$, $A^{\boxtimes^0} = \mathbb{I}$ and $A^k = A^{k-1} \times A$, $A^{\boxtimes^k} = A^{\boxtimes^{k-1}} \boxtimes A$, for all $k > 0$.

With these notions, a periodic matrix can be defined as follows.

Definition 20. *A matrix $A \in \mathbb{Z}_\infty^{m\times m}$ is called periodic if the sequence of tropical powers $\{A^{\boxtimes^k}\}_{k=1}^{\infty}$ is periodic.*

Intuitively, if A is the incidence matrix of a weighted digraph, then the sequence $\{A^{\boxtimes^k}\}_{k=1}^{\infty}$ of tropical powers of A gives the minimal weight paths of lengths $k = 1, 2, \ldots$ between any two vertices of the graph. It has been proved in [52] that every matrix $A \in \mathbb{Z}_\infty^{m\times m}$ is periodic. We define the prefix (period) of a matrix A as the prefix (period) of the sequence $\{A^{\boxtimes^k}\}_{k=1}^{\infty}$. We will often refer to periodicity (or prefix, period) of graphs, by which we mean periodicity (or prefix, period, respectively) of the corresponding incidence matrix.

We have argued that the complexity of the transitive closure algorithm depends on the size of the prefix and the period of the input relations, which will be later (in Section 7.2) shown to be bounded by the size of the prefix and the period of a certain kind of graphs, for difference bounds and octagonal relations. The result of [52] is however not suitable to establish bounds on the prefix and period of a graph. In this section, we therefore give an alternative proof of periodicity of matrices that moreover establishes bounds on sizes of their prefix and period.

Recall from Section **??** that given a path π, we denote by $w(\pi)$ its weight and that $\overline{w}(\pi)$ denotes its average weight. If $\lambda_1, \ldots, \lambda_k$ are pairwise distinct elementary cycles, the expression $\theta = \sigma_1 \cdot \lambda_1^* \cdot \sigma_2 \ldots \sigma_k \cdot \lambda_k^* \cdot \sigma_{k+1}$ is called a *path scheme of size* k. A path scheme $\theta = \sigma_1.\lambda_1^*.\sigma_2$ such that $|\sigma_1.\sigma_2| \leq |V|^4$ is called *basic*. A path scheme encodes

[3] Actually, the dual structure $(\mathbb{Z}_{-\infty}, \max, +, -\infty, 0)$ is also known as the tropical semiring in the literature.

the infinite set of paths $[\![\theta]\!] = \{\sigma_1 \cdot \lambda_1^{n_1} \cdot \sigma_2 \ldots \sigma_k \cdot \lambda_k^{n_k} \cdot \sigma_{k+1} \mid n_1, \ldots, n_k \in \mathbb{N}\}$. Given a weighted digraph $G = \langle V, E, w \rangle$, we denote by M_G its incidence matrix.

The first observation is that, for every minimal weight path in a weighted graph, there is an equivalent path following a path scheme whose size is bounded by the square of the size of the graph.

Proposition 9. *Let $G = \langle V, E, w \rangle$ be a weighted digraph and ρ be a minimal weight path in G. Then there exists a path scheme $\theta = \sigma_1 \cdot \lambda_1^* \cdot \ldots \cdot \sigma_k \cdot \lambda_k^* \cdot \sigma_{k+1}$ in G, such that $\sigma_1, \ldots, \sigma_{k+1}$ are acyclic and $k \leq \|V\|^2$, and a path $\rho' \in [\![\theta]\!]$ starting and ending in the same vertices as ρ, such that $|\rho| = |\rho'|$ and $w(\rho) = w(\rho')$.*

Proof: For each vertex $v \in V$, we partition the set of elementary cycles that start and end in v, according to their length. The representative of each equivalence class is chosen to be a cycle of minimal weight in the class. Since the length of each elementary cycle is at most $\|V\|$, there are at most $\|V\|^2$ such equivalence classes.

Let ρ be any path of minimal weight in G. First, notice that ρ can be factorized as:

$$\rho = \sigma_1 \cdot \lambda_1 \cdot \ldots \cdot \sigma_k \cdot \lambda_k \cdot \sigma_{k+1}$$

where $\sigma_1, \ldots, \sigma_{k+1}$ are elementary acyclic paths, and $\lambda_1, \ldots, \lambda_k$ are elementary cycles. This factorization can be achieved by a traversal of ρ while collecting the vertices along the way in a bag. The first vertex which is already in the bag marks the first elementary cycle. Then we empty the bag and continue until the entire path is traversed.

Next, we repeat the following two steps until nothing changes:

1. For all $i = 1, \ldots, k-1$ move all cycles λ_j, $j > i$, starting and ending with the same vertex as λ_i, next to λ_i, in the ascending order of their lengths. The result is a path ρ' of the same length and weight as ρ.
2. Factorize any remaining non-elementary acyclic path $\sigma_i \cdot \sigma_{i+1} \cdot \ldots \cdot \sigma_{i+j}$ as in the previous.

The loop above is shown to terminate, since the sum of the lengths of the remaining acyclic paths decreases with every iteration. The result is a path of the same length and weight as ρ, which starts and ends in the same vertices as ρ, in which all elementary cycles of the same length are grouped together. Since $w(\rho)$ is minimal for $|\rho|$, same holds for ρ', and moreover, all elementary cycles can be replaced by their equivalence class representatives, without changing neither the length, nor the weight of the path. The result is a path which belongs to a scheme with at most $\|V\|^2$ cycles. □

Second, for every minimal weight path in the graph, there exists an equivalent path which follows a basic path scheme.

Lemma 6. *Let $G = \langle V, E, w \rangle$ be a weighted digraph and ρ be a minimal weight path. Then there exists a path ρ', starting and ending in the same vertices as ρ, such that $w(\rho) = w(\rho')$ and $|\rho| = |\rho'|$, and a basic path scheme $\theta = \sigma \cdot \lambda^* \cdot \sigma'$ such that $\rho' \in [\![\theta]\!]$.*

Proof: By Proposition 9, for any path ρ in G there exists a path scheme $\theta = \sigma_1 \cdot \lambda_1^* \cdot \sigma_2 \ldots \sigma_k \cdot \lambda_k^* \cdot \sigma_{k+1}$, such that $\sigma_1, \ldots, \sigma_{k+1}$ are acyclic and $k \leq \|V\|^2$, and a path

ρ', starting and ending in the same vertices as ρ, of the same weight and length as ρ, such that $\rho' = \sigma_1 \cdot \lambda_1^{n_1} \cdot \sigma_2 \ldots \sigma_k \cdot \lambda_k^{n_k} \cdot \sigma_{k+1}$ for some $n_1, \ldots, n_k \geq 0$. Suppose that λ_i is a cycle with minimal average weight among all cycles in the scheme, i.e. $\frac{w(\lambda_i)}{|\lambda_i|} \leq \frac{w(\lambda_j)}{|\lambda_j|}$, for all $1 \leq j \leq k$. For each n_j there exist $p_j \geq 0$ and $0 \leq q_j < |\lambda_i|$, such that $n_j = p_j \cdot |\lambda_i| + q_j$. Let ρ' be the path:

$$\sigma_1 \cdot \lambda_1^{q_1} \cdot \sigma_2 \ldots \sigma_{i-1} \cdot \lambda_i^{n_i + \sum_{j=1}^{i-1} p_j \cdot |\lambda_j| + \sum_{j=i+1}^{k} p_j \cdot |\lambda_j|} \cdot \sigma_{i+1} \cdot \ldots \sigma_k \cdot \lambda_k^{q_k} \cdot \sigma_{k+1}$$

It is easy to check that $|\rho'| = |\rho|$ and $w(\rho') \leq w(\rho)$.

Clearly ρ' follows the path scheme $\rho_1 \cdot \lambda_i^* \cdot \rho_2$, where $\rho_1 = \sigma_1 \cdot \lambda_1^{q_1} \cdot \sigma_2 \ldots \sigma_{i-1}$ and $\rho_2 = \sigma_{i+1} \cdot \ldots \sigma_k \cdot \lambda_k^{q_k} \cdot \sigma_{k+1}$. Since $\lambda_1, \ldots, \lambda_k$ are elementary paths, all their lengths are strictly smaller than $\|V\|$. Since $q_j < |\lambda_i| \leq \|V\|$, and $k \leq \|V\|^2$, by Proposition 9, we have that $|\rho_1 \cdot \rho_2| < \|V\|^4$. Thus, $\rho_1 \cdot \lambda_i^* \cdot \rho_2$ is basic. □

The following lemma shows that, for a sufficiently long minimal weight path, there exists an equivalent path which follows a basic path scheme and moreover, this path scheme is followed by infinitely many minimal paths. Recall that

$$\mu(G) = \max\{|n| \mid u \xrightarrow{n} v \text{ in } G\}$$

denotes the maximum absolute value of all weights in G.

Lemma 7. *Let $G = \langle V, E, w \rangle$ be a weighted digraph, and $u, v \in V$ be two vertices. Then for every minimal weight path ρ form u to v, such that $|\rho| \geq \mu(G) \cdot \|V\|^6$, there exists a path ρ' from u to v, such that $w(\rho) = w(\rho')$ and $|\rho| = |\rho'|$, and a basic path scheme $\theta = \sigma \cdot \lambda^* \cdot \sigma'$, such that $\rho' = \sigma \cdot \lambda^b \cdot \sigma'$, for some $b \geq 0$. Moreover, there exists $c \mid \frac{lcm(1,\ldots,\|V\|-1)}{|\lambda|}$ such that $\sigma \cdot \lambda^{b+kc} \cdot \sigma'$ is a minimal weight path from u to v, for all $k \geq 0$.*

Proof: By Lemma 6, every minimal weight path from u to v follows a basic path scheme. Let $L > 0$ be an integer, and let $\sigma_i \cdot \lambda_i^* \cdot \sigma_i'$ and $\sigma_j \cdot \lambda_j^* \cdot \sigma_j'$ be two possible path schemes such that $\rho_i = \sigma_i \cdot \lambda_i^{b_i} \cdot \sigma_i'$, $\rho_j = \sigma_j \cdot \lambda_j^{b_j} \cdot \sigma_j'$ are two paths of length L, for some $b_i, b_j \geq 0$. We assume without loss of generality that λ_i has smaller average weight, i.e. $\overline{w}(\lambda_i) < \overline{w}(\lambda_j)$. We first prove that, if $L \geq \mu(G) \cdot \|V\|^6$, then $w(\rho_i) \leq w(\rho_j)$. We have:

$$\begin{aligned} b_i &= \tfrac{L - |\sigma_i \cdot \sigma_i'|}{|\lambda_i|}, \\ b_j &= \tfrac{L - |\sigma_j \cdot \sigma_j'|}{|\lambda_j|}, \end{aligned}$$

and

$$\begin{aligned} w(\rho_i) &= w(\sigma_i \cdot \sigma_i') + \tfrac{L - |\sigma_i \cdot \sigma_i'|}{|\lambda_i|} w(\lambda_i), \\ w(\rho_j) &= w(\sigma_j \cdot \sigma_j') + \tfrac{L - |\sigma_j \cdot \sigma_j'|}{|\lambda_j|} w(\lambda_j). \end{aligned}$$

Then $w(\rho_i) \leq w(\rho_j)$ if and only if

$$L \geq \frac{|\lambda_i||\lambda_j|(w(\sigma_i \cdot \sigma_i') - w(\sigma_j \cdot \sigma_j')) + |\lambda_i||\sigma_j \cdot \sigma_j'|w(\lambda_j) - |\lambda_j||\sigma_i \cdot \sigma_i'|w(\lambda_i)}{w(\lambda_j)|\lambda_i| - w(\lambda_i)|\lambda_j|}.$$

Since λ_i has a strictly smaller average weight than λ_j, we have that $w(\lambda_j)|\lambda_i| - w(\lambda_i)|\lambda_j| > 0$ and since $w(\lambda_i), w(\lambda_j), |\lambda_i|, |\lambda_j| \in \mathbb{Z}$, we have that $w(\lambda_j)|\lambda_i| - w(\lambda_i)|\lambda_j| \geq 1$. By Lemma 6, we have $|\sigma_i \cdot \sigma_i'|, |\sigma_j \cdot \sigma_j'| \leq \|V\|^4$, and $w(\sigma_i \cdot \sigma_i') - w(\sigma_j \cdot \sigma_j') \leq \mu(G) \cdot \|V\|^4$. We compute:

$$\begin{array}{rcl} L & \geq & \mu(G) \cdot \|V\|^6 \\ & \geq & |\lambda_i||\lambda_j|(w(\sigma_i \cdot \sigma_i') - w(\sigma_j \cdot \sigma_j')) + |\lambda_i||\sigma_j \cdot \sigma_j'|w(\lambda_j) - |\lambda_j||\sigma_i \cdot \sigma_i'|w(\lambda_i) \\ & \geq & \frac{|\lambda_i||\lambda_j|(w(\sigma_i \cdot \sigma_i') - w(\sigma_j \cdot \sigma_j')) + |\lambda_i||\sigma_j \cdot \sigma_j'|w(\lambda_j) - |\lambda_j||\sigma_i \cdot \sigma_i'|w(\lambda_i)}{w(\lambda_j)|\lambda_i| - w(\lambda_i)|\lambda_j|} \end{array}$$

Since the choice of ρ_i and ρ_j was arbitrary, for each $L \geq \mu(G) \cdot \|V\|^6$, the path scheme with minimal average weight cycle is chosen by the minimal weight path of length L.

Second, we show that this happens periodically. For two paths $\rho_i = \sigma_i \cdot \lambda_i^{b_i} \cdot \sigma_i'$ and $\rho_j = \sigma_j \cdot \lambda_j^{b_j} \cdot \sigma_j'$ of equal lengths, as before, let $c_{ij} = lcm(|\lambda_i|, |\lambda_j|)$, $c_i = \frac{c_{ij}}{|\lambda_i|}$ and $c_j = \frac{c_{ij}}{|\lambda_j|}$. We have that $|\lambda_i^{kc_i}| = |\lambda_j^{kc_j}| = kc_{ij}$, for all $k \geq 0$. Moreover, since $\overline{w}(\lambda_i) < \overline{w}(\lambda_j)$, we have that $w(\lambda_i^{kc_i}) < w(\lambda_j^{kc_j})$. It follows that

$$w(\sigma_i \cdot \lambda_i^{b_i + kc_i} \cdot \sigma_i') \leq w(\sigma_j \cdot \lambda_j^{b_j + kc_j} \cdot \sigma_j')$$

for all $k \geq 0$. Finally, since $|\lambda_i|, |\lambda_j| < \|V\|$, we have that $c_{ij} \mid lcm(1, \ldots, \|V\|-1)$ and thus $c_i \mid \frac{lcm(1,\ldots,\|V\|-1)}{|\lambda_i|}$. Since the choice of i does not change this fact, it is enough to take $c = \frac{lcm(1,\ldots,\|V\|-1)}{|\lambda|}$ and $b = b_i$ to obtain that $\sigma_i \cdot \lambda_i^{b+kc} \cdot \sigma_i'$ has minimal weight among all paths from u to v of the same length, for all $k \geq 0$. □

The following lemma is essential to prove an upper bound on the period of weighed digraphs.

Lemma 8. *For each $n \geq 1$, $lcm(1, \ldots, n)$ is bounded by $2^{\mathcal{O}(n)}$.*

Proof: We know that $lcm(1, \ldots, n) = \prod_{p \leq n} p^{\lfloor log_p(n) \rfloor}$ where the product is taken only over primes p. Obviously, for every prime p we have that $p^{\lfloor log_p(n) \rfloor} \leq p^{log_p(n)} = n$. Hence, $lcm(1, \ldots, n) \leq \prod_{p \leq n} n = n^{\pi(n)}$, where $\pi(n)$ denotes the prime-counting function (which gives the number of primes less than or equal to n, for every natural number n). Using the prime number theorem which states that $\lim_{n \to \infty} \frac{\pi(n)}{n/ln(n)} = 1$ we can effectively bound $\pi(n)$. That is, for any $\epsilon > 0$, there exists n_ϵ such that $\frac{\pi(n)}{n/ln(n)} \leq (1 + \epsilon)$ for all $n \geq n_\epsilon$. Consequently, $n^{\pi(n)} \leq n^{(1+\epsilon)n/ln(n)} = e^{(1+\epsilon)n} = 2^{log_2(e)(1+\epsilon)n} = 2^{\mathcal{O}(n)}$ for all $n \geq n_\epsilon$, and completes the proof. □

The following theorem gives asymptotic bounds on the size of the prefix and the period of a weighted digraph $G = \langle V, E, w \rangle$, in terms of $\mu(G)$ and $\|V\|$.

Theorem 6. *Let $G = \langle V, E, w \rangle$ be a digraph, and $M_G \in \mathbb{Z}_\infty^{\|V\| \times \|V\|}$ be its incidence matrix. Then, the sequence $\{M_G^{\boxtimes^i}\}_{i \geq 0}$ is periodic. Moreover, its prefix b is bounded by $\mu(G) \cdot \mathcal{O}(\|V\|^6)$, and its period divides $\mathrm{lcm}(1, \ldots, \|V\|-1)$ and is bounded by $2^{\mathcal{O}(\|V\|)}$.*

Proof: A direct consequence of Lemma 7 is that each minimal weight path ρ in G of length at least $\mu(G) \cdot \|V\|^6$ must be of the form $\rho = \sigma \cdot \lambda^b \cdot \sigma'$, and moreover, for

some $c \mid \frac{lcm(1,\ldots,\|V\|-1)}{|\lambda|}$, we have that $\sigma \cdot \lambda^{b+kc} \cdot \sigma'$ is a minimal weight path, for all $k \geq 0$. Hence, for each $1 \leq i, j \leq \|V\|$, the sequence $\{(M_G^{\boxtimes k})_{ij}\}_{k=0}^{\infty}$ is periodic with prefix at most $\mu(G) \cdot \|V\|^6$ and period which divides $lcm(1, \ldots, \|V\| - 1)$. The prefix b of M_G is the maximum of all prefixes, and the period c is the least common multiple of the periods of the sequences $\{(M_G^{\boxtimes i})_{ij}\}_{k=0}^{\infty}$, respectively. Hence b is bounded by $\mu(G) \cdot \mathcal{O}(\|V\|^6)$ and c divides $\mathrm{lcm}(1, \ldots, \|V\| - 1)$. By Lemma 8, $lcm(1, \ldots, \|V\| - 1)$ is bounded by $2^{\mathcal{O}(\|V\|)}$. Since c divides $\mathrm{lcm}(1, \ldots, \|V\| - 1)$, the same bound for c follows immediately. □

7.2. Difference Bounds Relations

Recall from Section 6.1 that a difference bounds relation $R \subseteq \mathbb{Z}^N \times \mathbb{Z}^N$ can be equivalently represented as a difference bounds matrix (DBM) M_R. Similarly, for each DBM M there is a corresponding difference bounds relation Φ_M. Furthermore, difference bounds relations can be represented canonically by DBMs M_R^*.

The first step to proving that the class $\mathcal{R}_{db}$ is periodic is defining the mappings between relations and matrices (Definition 4). Given a consistent difference bounds relation $R \in \mathcal{R}_{db}$, we define $\sigma(R) = M_R^* \in \mathbb{Z}_{\infty}^{2N \times 2N}$ to be the closed characteristic DBM of R. Dually, for any DBM $M \in \mathbb{Z}_{\infty}^{2N \times 2N}$, let $\rho(M) = \Phi_M \in \mathcal{R}_{db}$ be the difference bounds relation corresponding to M. We clearly have $\rho(\sigma(R)) \Leftrightarrow R$, for all consistent relations R, as required by Definition 4.

In order to define a function $\pi : \mathbb{Z}[k]_{\infty}^{m \times m} \to \mathcal{R}[k]$ mapping matrices of linear terms of the form $\alpha \cdot k + \beta$, with integer coefficients, into parametric relations $R(k, \mathbf{x}, \mathbf{x}')$, we define the class of *parametric* difference bounds relations.

Definition 21. *A formula $\phi(\mathbf{x}, k)$ is a* parametric difference bounds constraint *if it is equivalent to a finite conjunction of atomic propositions of the form $x_i - x_j \leq t_{ij}$, for some $1 \leq i, j \leq N$, $i \neq j$, where t_{ij} are univariate linear terms in k.*

The class of parametric difference bounds relations with parameter k is denoted as $\mathcal{R}_{db}[k]$. Similar to the non-parametric case (Definition 8), a parametric difference bounds constraint $\phi(k)$ can be represented by a matrix $M_\phi[k] \in \mathbb{Z}[k]_{\infty}^{N \times N}$ of univariate linear terms, where $(M_\phi[k])_{ij} = t_{ij}$ if $x_i - x_j \leq t_{ij}$ occurs in ϕ, and ∞ otherwise. Dually, a matrix $M[k]$ of linear terms corresponds to the formula $\Phi_M(k) \Leftrightarrow \bigwedge_{M[k]_{ij} \neq \infty} x_i - x_j \leq M[k]_{ij}$. With these considerations, we define $\pi(M[k]) = \Phi_M(k)$ to be the parametric counterpart of the ρ function from Definition 4. Clearly, for each matrix $M \in \mathbb{Z}[k]_{\infty}^{m \times m}$, the mapping π satisfies the required property that $\pi(M)[n/k] \Leftrightarrow \rho(M[n])$ for all $n \in \mathbb{Z}$.

7.3. Proving Periodicity

In this section, we prove that all relations defined using difference bounds constraints are periodic in the sense of Definition 4. A direct consequence is that these relations are also periodic, which ensures the termination of Algorithm 5 on the $\mathcal{R}_{db}$ class.

Let $R \in \mathcal{R}_{db}$ be an arbitrary difference bounds relation for the rest of this section. If R is not $*$-consistent, then by Definition 4, R is periodic. We consider from now on

that R is $*$-consistent and prove that the sequence $\{\sigma(R^i)\}_{i=0}^{\infty}$ is periodic in this case. We have $\sigma(R^i) = M^*_{R^i}$ for any $i \geq 0$.

The proof idea is that the entries of the sequence $\{M^*_{R^i}\}_{i=0}^{\infty}$ represent minimal weight paths in the graph corresponding to the i-times "unfolding" of R, for any $i \geq 0$. These paths form a regular language recognized by a finite weighted automaton. Consequently, the minimal weights for $i = 0, 1, 2, \ldots$ are entries in the sequence of tropical powers of the incidence matrix of this automaton. But then they form periodic sequences, according to Theorem 6.

For all $1 \leq i, j \leq N$, we obtain the following equalities:

$$\begin{array}{lcl} [\sigma(R^m)]_{i,j} & = & \min\{x_i^0 \rightarrow x_j^0\} \\ [\sigma(R^m)]_{i+N,j+N} & = & \min\{x_i^m \rightarrow x_j^m\} \\ [\sigma(R^m)]_{i,j+N} & = & \min\{x_i^0 \rightarrow x_j^m\} \\ [\sigma(R^m)]_{i+N,j} & = & \min\{x_i^m \rightarrow x_j^0\} \end{array} \tag{6}$$

Recall the definition of the *zigzag automata* from Section 6.1.3 that recognize paths within constraint graphs. In the following, we view these automata as reasoning tools, needed to prove the periodicity of the difference bounds constraints. Recall that $T_R = \langle Q, \Delta, w \rangle$ is the common transition table of all zigzag automata for $R \in \mathcal{R}_{db}$. Let $\mathcal{M}_R \in \mathbb{Z}_{\infty}^{\|Q\| \times \|Q\|}$ be the incidence matrix of T_R, where $\|Q\|$ is the number of control states in T_R. Without loss of generality, we assume that states in Q are both reachable and co-reachable[4]. For each pair of variables x_i, x_j, there are eight indices, denoted as $I^{ef}_{i,j}$, F^{ef}, I^{eb}, $F^{eb}_{i,j}$, I^{of}_i, F^{of}_j, I^{ob}_i, $F^{ob}_j \in \{1, \ldots, \|Q\|\}$ corresponding to the initial and final states of the four zigzag automata, respectively. According to Lemma 5, the minimal weight path of length $m + 2$ from $I^{ef}_{i,j}$ to F^{ef} matches the minimal weight path between the extremal points x_i^0 and x_j^0 of $\mathcal{G}_R^m$. Similarly for paths from I^{eb} to $F^{eb}_{i,j}$, from I^{of}_i to F^{of}_j, and from I^{ob}_i to F^{ob}_j. However the weights of the paths in the zigzag automata are captured by the tropical powers of $\mathcal{M}_R$, as follows:

$$\begin{array}{rcl} \min\{x_i^0 \rightarrow x_j^0\} & = & \left[{\mathcal{M}_R}^{\boxtimes^{m+2}}\right]_{I^{ef}_{i,j}, F^{ef}} \\ \min\{x_i^m \rightarrow x_j^m\} & = & \left[{\mathcal{M}_R}^{\boxtimes^{m+2}}\right]_{I^{eb}, F^{eb}_{i,j}} \\ \min\{x_i^0 \rightarrow x_j^m\} & = & \left[{\mathcal{M}_R}^{\boxtimes^{m+2}}\right]_{I^{of}_i, F^{of}_j} \\ \min\{x_i^m \rightarrow x_j^0\} & = & \left[{\mathcal{M}_R}^{\boxtimes^{m+2}}\right]_{I^{ob}_i, F^{ob}_j} \end{array} \tag{7}$$

By Theorem 6, the tropical powers of $\mathcal{M}_R$ form a periodic sequence, therefore the sequence $\{{\mathcal{M}_R}^{\boxtimes^{m+2}}\}_{m \geq 0}$ is periodic. By equating the equivalences (6) and (7) from the previous, we obtain that the sequence $\{\sigma(R^m)\}_{m \geq 0}$ is periodic as well. The following theorem summarizes the above arguments.

Theorem 7. *The class of difference bounds relations is periodic.*

[4]A state is said to be *reachable* if there exists a path from an initial state to it, and *co-reachable* if there exists a path from it to a final state.

Moreover, since M_{R^m} is a projection of $\mathcal{M}_R^{\boxtimes^{m+2}}$ for all $m \geq 0$ if R is $*$-consistent, the prefix of a $*$-consistent relation R is bounded by the prefix of T_R. Similar claim can be made for the period of R.

Proposition 10. *Let $T_R = (Q, \Delta, w)$ be the common transition table of zigzag automata defined for a $*$-consistent difference bounds relation R. Then, the size of the prefix of R is bounded by the size of the prefix of T_R and the period of R divides the period of T_R.*

Proof: The proof of Lemma 1 shows that if b_{ij} (c_{ij}) is the prefix (period) of the sequence $\{(\mathcal{M}_R^{\boxtimes^m})_{ij}\}_{m\geq 0}$ for all $1 \leq i, j \leq \|Q\|$, then the sequence $\{\mathcal{M}_R^{\boxtimes^m}\}_{m\geq 0}$ has the prefix defined as $b = \max_{ij}\{b_{ij}\}$ and the period defined as $c = \mathrm{lcm}_{ij}\{c_{ij}\}$. Since R is $*$-consistent, M_{R^m} is a projection of $\mathcal{M}_R^{\boxtimes^{m+2}}$ for all $m \geq 0$. Then the bounds stated in this proposition follow from the definition of b and c. □

In conclusion, Algorithm 5 will terminate on difference bounds relations. Moreover, the result is formula definable in Presburger arithmetic. In particular, this also simplifies the proof that transitive closures of difference bounds relations are Presburger definable, from [15]. The following result is needed in the following section to design a cost-effective implementation of Algorithm 5.

Corollary 1. *If $R \in \mathcal{R}_{db}$ be a difference bounds relation, the rate Λ of the periodic sequence $\{\sigma(R^i)\}_{i=0}^{\infty}$ is a closed DBM.*

Proof: Since $\sigma(R^m) = M^*_{R^m}$ for all $m \geq 0$, $\sigma(R^m)$ is closed and thus we have for all $1 \leq i, j, k \leq 2N$ and $m \geq 0$:

$$\left[\sigma(R^m)\right]_{i,j} \leq \left[\sigma(R^m)\right]_{i,k} + \left[\sigma(R^m)\right]_{k,j}$$

Since $\{\sigma(R^m)\}_{m=0}^{\infty}$ is periodic, there exists $b \geq 0$, $c > 0$ and $\Lambda \in \mathbb{Z}_{\infty}^{2N\times 2N}$ such that:

$$\sigma(R^{b+nc}) = \sigma(R^b) + n \cdot \Lambda$$

for all $n \geq 0$. Consequently, we have, for all $1 \leq i, j, k \leq 2N$ and all $n \geq 0$:

$$\left[\sigma(R^b)\right]_{i,j} + n \cdot \Lambda_{ij} \leq \left[\sigma(R^b)\right]_{i,k} + \left[\sigma(R^b)\right]_{k,j} + n \cdot (\Lambda_{ik} + \Lambda_{kj})$$

We obtain:

$$n \cdot (\Lambda_{ij} - \Lambda_{ik} - \Lambda_{kj}) \leq \left[\sigma(R^b)\right]_{i,k} + \left[\sigma(R^b)\right]_{k,j} - \left[\sigma(R^b)\right]_{i,j},\ \forall n \geq 0$$

Suppose that Λ is not closed i.e., there exist $1 \leq i, j, k \leq 2N$ such that $\Lambda_{ij} > \Lambda_{ik} + \Lambda_{kj}$, we get a contradiction with the above. □

7.4. Checking $*$-consistency and Periodicity

In this section, we describe cost-effective ways to implement the MAXCONSISTENT and MAXPERIODIC procedures from Algorithm 5 for difference bounds relations.

First, we need to introduce several technical notions. A *univariate linear term* is a term of the form $\alpha \cdot k + \beta$, where $\alpha, \beta \in \mathbb{Z}$ are integer constants. Let $\mathbb{Z}[k]$ denote the set of all univariate linear terms with variable k. For two sets $S, T \subseteq \mathbb{Z}[k]$, we define $S \oplus T = \{(\alpha_1 + \alpha_2) \cdot k + \beta_1 + \beta_2 \mid \alpha_1 \cdot k + \beta_1 \in S,\ \alpha_2 \cdot k + \beta_2 \in T\}$. For two linear terms $t_1 = \alpha_1 \cdot k + \beta_1$ and $t_2 = \alpha_2 \cdot k + \beta_2$, we define the partial order on terms $t_1 \preceq t_2 \Leftrightarrow \alpha_1 \leq \alpha_2 \ \wedge \ \beta_1 \leq \beta_2$. We denote the strict inequality on terms by $t_1 \prec t_2 \Leftrightarrow t_1 \preceq t_2 \ \wedge \ t_2 \not\preceq t_1$. For a finite set of linear terms S, we denote by $\textsc{MinTerms}(S) = \{t \in S \mid \forall s \in S \,.\, s \not\prec t\}$ the set of minimal terms in S, with respect to this order. For a set of integer constants $\{a_1, \ldots, a_n\}$, we denote by $\textsc{MaxMin}\{a_i\}_{i=1}^n$ the positive value $\max\{a_i\}_{i=1}^n - \min\{a_i\}_{i=1}^n$.

Proposition 11. *Let $S = \{\alpha_i \cdot k + \beta_i\}_{i=1}^m$ be a set of univariate linear terms. Then*

$$\|\textsc{MinTerms}(S)\| \leq \min(\textsc{MaxMin}\{\alpha_j\}_{j=1}^m, \textsc{MaxMin}\{\beta_j\}_{j=1}^m)$$

Proof: A term $\alpha \cdot k + \beta$ corresponds to the point (α, β) in the 2-dimensional space. All terms from S are points in the rectangle defined by the bottom left corner $(\min\{\alpha_j\}_{j=1}^m, \min\{\beta_j\}_{j=1}^m)$ and the upper right corner $(\max\{\alpha_j\}_{j=1}^m, \max\{\beta_j\}_{j=1}^m)$. Since all terms in $\textsc{MinTerms}(S)$ are incomparable w.r.t. $\preceq$, there can be at most $\min(\max\{\alpha_j\}_{j=1}^m - \min\{\alpha_j\}_{j=1}^m, max\{\beta_j\}_{j=1}^m - \min\{\beta_j\}_{j=1}^m)$ such terms. Hence the result. □

Given a linear term $t = \alpha \cdot k + \beta$, we denote by $t(n)$ the value $\alpha \cdot n + \beta$, for any $n \in \mathbb{N}$. The set of valuations of a term t, with respect to the threshold ℓ is $[\![t]\!]_{\geq \ell} = \{t(n) \mid n \geq \ell\}$. These notations are naturally lifted to sets of terms, i.e. $T(n) = \{t(n) \mid t \in T\}$. and $[\![T]\!]_{\geq \ell} = \bigcup_{t \in T} [\![t]\!]_{\geq \ell}$.

Unlike DBMs with constant entries, parametric DBMs do not have a closed form, since in general, the minimum of two univariate linear terms cannot be defined again as a linear term. A way around this problem is using matrices of sets of univariate linear terms, with the convention that a set $T = \{t_1(k), \ldots, t_n(k)\}$ of univariate linear terms denotes the function $k \mapsto min\{t_1, \ldots, t_n\}$, and $\min(\emptyset) = \infty$. The Floyd-Warshall algorithm for computing closed forms of DBMs with constant entries can be easily adapted to parametric DBMs.

Algorithm 7 takes as input a matrix of univariate linear terms, and produces a matrix of sets of such terms (each set of terms T is interpreted as $\min(T)$). Lines 2-7 initialize the output matrix with sets of terms. Lines 8-14 correspond to the classical Floyd-Warshall iteration.

Proposition 12. *Let $M \in \mathbb{Z}_\infty^{m \times m}[k]$ be a parametric DBM, such that $M_{ij} = \alpha_{ij} \cdot k + \beta_{ij}$, for all $1 \leq i, j \leq m$. Then, Algorithm 7 runs in at most $\mathcal{O}(\mu^3 \cdot m^6)$ time where*

$$\mu = \min(\max_{1 \leq i,j \leq m}\{|\alpha_{ij}|\}, \max_{1 \leq i,j \leq m}\{|\beta_{ij}|\})$$

Moreover, we have $\|\mathcal{M}_{ij}\| \leq 2m \cdot \mu$.

Proof: Each term $\alpha \cdot k + \beta \in \mathcal{M}[i][j]$ is a sum of at most m terms $\alpha_{ij} \cdot k + \beta_{ij}$. We have:

$$\begin{array}{rcl} -m \cdot \max_{1 \leq i,j \leq m}\{|\alpha_{ij}|\} & \leq & \alpha \leq m \cdot \max_{1 \leq i,j \leq m}\{|\alpha_{ij}|\}, \text{ and} \\ -m \cdot \max_{1 \leq i,j \leq m}\{|\beta_{ij}|\} & \leq & \beta \leq m \cdot \max_{1 \leq i,j \leq m}\{|\beta_{ij}|\}. \end{array}$$

Algorithm 7 Closure Algorithm for Parametric DBMs

```
1: procedure ParametricFW(M)
2:     for all i = 0, ..., m − 1 do
3:         for all j = 0, ..., m − 1 do
4:             if M[i][j] = ∞ then
5:                 M[i][i] ← ∅
6:             else
7:                 M[i][j] ← {M[i][j]}
8:     for all k = 0, ..., m − 1 do
9:         for all i = 0, ..., m − 1 do
10:            for all j = 0, ..., m − 1 do
11:                T_0 ← M[i][j]
12:                T_1 ← M[i][k]
13:                T_2 ← M[k][j]
14:                M[i][j] ← MinTerms (T_0 ∪ (T_1 ⊕ T_2))
15:    return M[i][j]
16: procedure MinTerms(S)
17:    return {t ∈ S | ∀s ∈ S . s ⊀ t}
```

By an argument similar to the one used in Proposition 11, we have that $\|\mathcal{M}[i][j]\| \leq 2m \cdot \mu$, where μ is defined as $\mu = \min(\max_{1\leq i,j\leq m}\{|\alpha_{ij}|\}, \max_{1\leq i,j\leq m}\{|\beta_{ij}|\})$. Therefore, each call to MinTerms takes at most $\mathcal{O}(m^3 \cdot \mu^3)$ time. Since the classical Floyd-Warshall algorithm (i.e. Algorithm 7 in which we consider that MinTerms needs constant time) runs in time $\mathcal{O}(m^3)$, we obtain the result. □

MaxConsistent.. Given a difference bounds relation R, integers $b \geq 0, c > 0$ such that R^{b+2c} is consistent, and a matrix $\Lambda \in \mathbb{Z}_\infty^{2N\times 2N}$, let us denote $M_{R,b,\Lambda} = k \cdot \Lambda + \sigma(R^b) \in \mathbb{Z}[k]_\infty^{2N\times 2N}$. With this notation, we have:

$$\textsc{MaxConsistent}(R, b, \Lambda) = \sup\{n \in \mathbb{N} \mid M_{R,b,\Lambda}[n] \text{ is consistent}\}.$$

Since R^{b+2c} is consistent, it follows that $\textsc{MaxConsistent}(R, b, \Lambda) > 2$ and hence, we can define $\textsc{MaxConsistent}(R, b, \Lambda)$ equivalently as

$$\textsc{MaxConsistent}(R, b, \Lambda) = \inf\{n \in \mathbb{N} \mid M_{R,b,\Lambda}[n] \text{ is inconsistent}\} - 1.$$

In analogy to the non-parametric case, the inconsistency of a parametric difference bounds constraint amounts to the existence of a strictly negative elementary cycle in the constraint graph corresponding to $M_{R,b,\Lambda}[n]$ for some valuation $n \in \mathbb{N}$ of k. The MaxConsistent procedure can be implemented as follows. Let

$$\mathcal{M} = \textsc{ParametricFW}(M_{R,b,\Lambda})$$

as returned by Algorithm 7. Obviously, $M_{R,b,\Lambda}[n]$ is not consistent if and only if $\min(\mathcal{M}_{ii}[n]) < 0$ for some $i = 1, \ldots, 2N$. The minimal value of n for which this is the case is $K' = \min\{\Gamma(\mathcal{M}_{ii})\}_{i=1}^{2N}$, where Γ is a constant defined in the following lemma. Then, MaxPeriodic returns integer K defined as $K = K' - 1$.

Lemma 9. *Let $T = \{\alpha_i \cdot k + \beta_i\}_{i=1}^m$ be a set of univariate linear terms and $\ell \in \mathbb{N}$ be a constant. Then $\min_{n \geq \ell} T(n) < 0$ if and only if there exists $1 \leq i \leq m$ such that either $\alpha_i < 0$, or $\alpha_i \cdot \ell + \beta_i < 0$. Moreover the smallest value n such that $\min_{n \geq \ell}\{\alpha_i \cdot n + \beta_i\}_{i=1}^m < 0$ is $\Gamma(T) = \min_{j=1}^m \gamma_j$ where:*

$$\gamma_j = \begin{cases} \max(\ell, \lfloor -\frac{\beta_j}{\alpha_j} \rfloor + 1) & \text{if } \alpha_j < 0 \\ \ell & \text{if } \alpha_j \geq 0 \ \wedge \ \alpha_j \cdot \ell + \beta_j < 0 \\ \infty & \text{otherwise} \end{cases}$$

Proof: $\min_{n \geq \ell} T(n) < 0$ iff there exists $1 \leq i \leq m$ such that $\alpha_i \cdot n + \beta_i < 0$. Let us fix i for the rest of the proof. There are three cases:

- if $\alpha_i < 0$ we have $n \geq \lfloor -\frac{b_i}{\alpha_i} \rfloor + 1$, hence $n \geq \gamma_i = \max(\ell, \lfloor -\frac{b_i}{\alpha_i} \rfloor + 1)$.
- if $\alpha_i \geq 0$ and $\alpha_i \cdot \ell + \beta_i \geq 0$, we have $\alpha_i \cdot n + \beta_i \geq 0$, for all $n \geq \ell$, contradiction.
- else, we have $\alpha_i \geq 0$ and $\alpha_i \cdot \ell + \beta_i < 0$, in which case we have $n \geq \gamma_i = \ell$.

□

Proposition 13. *For a difference bounds relation R, integers $b \geq 0, c > 0$ such that R^b is consistent and a matrix $\Lambda \in \mathbb{Z}_\infty^{2N \times 2N}$,* MAXCONSISTENT*$(R, b, \Lambda)$ runs in time at most $\mathcal{O}((b+c)^3 \cdot \|R\|^3 \cdot N^9)$.*

Proof: Computing $\mathcal{M}$ requires one application of Algorithm 7. By Proposition 12, the call to Algorithm 7 requires time at most $\mathcal{O}(\mu^3 \cdot N^6)$, where:

$$\mu = \min(\max_{1 \leq i,j \leq 2N}\{|\Lambda_{ij}|\}, \max_{1 \leq i,j \leq 2N}\{|(\sigma(R^b))_{ij}|\})$$

Since the constraint graph $\mathcal{G}_R^b$ has $(b+1) \cdot N$ nodes, any minimal path between extremes may not exceed weight $(b+1) \cdot N \cdot \|R\|$. This is because R^b is consistent, i.e. there are no negative cycles in $\mathcal{G}_R^b$, and a path going through a positive cycle is not minimal. Since the rate Λ is computed as $\Lambda = \sigma(R^{b+c}) - \sigma(R^b)$, we similarly infer that $\Lambda_{ij} \leq (b+c+1) \cdot N \cdot \|R\|$ for all $1 \leq i, j \leq 2N$. Hence $\mu \leq (b+c+1) \cdot N \cdot \|R\|$, which gives the result. □

MAXPERIODIC.. Given a difference bounds relation R, integers $K \in \mathbb{N}_\infty$, $b \geq 0$ and $c > 0$, such that R^b is consistent, and a matrix $\Lambda \in \mathbb{Z}_\infty^{2N \times 2N}$, the procedure

$$\text{MAXPERIODIC}(R, b, \Lambda, c, K)$$

returns the maximal integer[5] $0 \leq n \leq K$ such that:

$$\forall 0 \leq \ell < n \,.\, \rho(\ell \cdot \Lambda + \sigma(R^b)) \circ R^c \Leftrightarrow \rho((\ell+1) \cdot \Lambda + \sigma(R^b))$$

[5]The successful test at line 8 of Algorithm 5 implies that $n \geq 2$.

or ∞, if $K = \infty$ and the above equivalence holds for all ℓ. The left-hand side of the equivalence can be encoded by a matrix of terms of the form $\min\{t_i\}_{i=1}^m$, where t_i are univariate linear terms, and can be computed by Algorithm 7. The DBM corresponding to the right-hand side is shown to be closed for all valuations of k, which means that the relation on the right-hand side of the equivalence can be defined simply by a parametric DBM, instead of a matrix of min-terms, which is the case for the left-hand side.

Lemma 10. *Let $R \in \mathcal{R}_{db}$ be a difference bounds relation, and Λ be the rate of the periodic sequence $\{\sigma(R^i)\}_{i=0}^{\infty}$. Then, for all $b \geq 0$ and $n > 0$, the DBM $n \cdot \Lambda + \sigma(R^b)$ is closed.*

Proof: A direct consequence of the fact that $\sigma(R^b)$ is closed by definition, and that Λ is also closed, by Corollary 1. □

We need thus to check equivalence (for all $k \geq 0$) between a matrix of minima of sets of linear terms in k and a parametric DBM. By Proposition 3, equivalence of two difference bounds constraints amounts to the equality of their closed DBMs. In order to find the maximal interval $0, \ldots, n$ in which $\min\{\alpha_i \cdot k + \beta_i\}_{i=1}^m = \alpha_0 \cdot k + \beta_0$ holds, for all $k = 0, \ldots, n$, we apply the following lemma to each entry in the left and right-hand side of the above equivalence, and return the minimal value among all entries, for which the equivalence holds, incremented by one[6].

Lemma 11. *Let $T = \{\alpha_i \cdot k + \beta_i\}_{i=1}^m$ be a set of univariate linear terms, $t_0 = \alpha_0 \cdot k + \beta_0 \in \mathbb{Z}[k]$ be a term, and $\ell \in \mathbb{N}$ be a constant. Then there exists an integer $\kappa > \ell + 1$ such that $\min T(n) = t_0(n)$, for all $\ell \leq n \leq \kappa$, if and only if the following hold:*

1. $\bigvee_{i=1}^m (\alpha_i = \alpha_0 \wedge \beta_i = \beta_0) \wedge \bigwedge_{i=1}^m \bigwedge_{j=0}^2 [\alpha_0 \cdot (\ell + j) + \beta_0 \leq \alpha_i \cdot (\ell + j) + \beta_i]$
2. $\kappa \leq \min\{\lfloor \frac{\beta_i - \beta_0}{\alpha_0 - \alpha_i} \rfloor \mid 1 \leq i \leq m,\ \alpha_0 \neq \alpha_i,\ \lfloor \frac{\beta_i - \beta_0}{\alpha_0 - \alpha_i} \rfloor > \ell + 1\}$

Proof: We assume without loss of generality that $m \geq 2$ and that all terms $\alpha_i \cdot k + \beta_i,\ i = 1, \ldots, m$ are distinct.
"⇒" If $\min T(n) = t_0(n)$, for all $\ell \leq n \leq \kappa$ and $\kappa > \ell + 1$, then clearly $\bigwedge_{i=1}^m \bigwedge_{j=0}^2 [\alpha_0 \cdot (\ell + j) + \beta_0 \leq \alpha_i \cdot (\ell + j) + \beta_i]$, i.e. the second conjunct of the first point is valid. To show the validity of the first conjunct of the first point, suppose without loss of generality that:

$$\begin{array}{rcccl} t_0(\ell) & = & t_1(\ell) & \leq & t_2(\ell) \\ t_0(\ell+1) & = & t_2(\ell+1) & \leq & t_1(\ell+1) \\ t_0(\ell+2) & < & t_1(\ell+2) & & \\ t_0(\ell+2) & < & t_2(\ell+2) & & \end{array}$$

The choice of t_1 and t_2 is not important. We obtain a contradiction in the following way:

$$\begin{array}{rcccl} \ell & = & \frac{\beta_1 - \beta_0}{\alpha_0 - \alpha_1} & \geq & \frac{\beta_2 - \beta_0}{\alpha_0 - \alpha_2} \\ \ell + 1 & = & \frac{\beta_2 - \beta_0}{\alpha_0 - \alpha_2} & \geq & \frac{\beta_1 - \beta_0}{\alpha_0 - \alpha_1} \end{array}$$

[6]Since $\forall 0 \leq \ell \leq \kappa\ .\ \phi$ if and only if $\forall 0 \leq \ell < \kappa + 1\ .\ \phi$.

To show the second point, assume by contradiction that $\kappa > \lfloor \frac{\beta_i - \beta_0}{\alpha_0 - \alpha_i} \rfloor > \ell + 1$ for some $i = 1, \ldots, m$, such that the term $\alpha_i \cdot k + \beta_i$ is distinct from $\alpha_0 \cdot k + \beta_0$. Since, by the first point, we have $\beta_0 \leq \beta_i$, and $\frac{\beta_i - \beta_0}{\alpha_0 - \alpha_i} > 0$, then $\alpha_0 > \alpha_i$. It follows that $\alpha_0 \cdot \kappa > \alpha_i \cdot \kappa + \beta_i$, which contradicts the fact that t_0 is minimal in the interval $\ell, \ldots, \kappa$. "$\Leftarrow$" By the first point, $t_0(n) = t_i(n)$, for all n, and $\min T(n) = t_0(n)$, for $n = \ell, \ell + 1, \ell + 2$. To prove that $\min T(n) = t_0(n)$ for all $\ell \leq n \leq \kappa$, assume by contradiction, that $\min T(p) = t_i(p) < t_0(p)$ for some $p = \ell, \ldots, \kappa$ and some $i = 1, \ldots, m$, such that t_i is distinct from t_0. But then we have $(\alpha_i - \alpha_0) \cdot p < \beta_0 - \beta_i \leq 0$. The last inequality is due to the first point. Since $p > 0$, we have that $\alpha_i < \alpha_0$, hence $\kappa \geq p > \lfloor \frac{\beta_i - \beta_0}{\alpha_0 - \alpha_i} \rfloor$, contradiction. □

Proposition 14. *For a difference bounds relation R, and integers $b \geq 0, c > 0$ such that R^{b+c} is consistent and a matrix $\Lambda \in \mathbb{Z}_{\infty}^{2N \times 2N}$,* MAXPERIODIC *runs in time at most $\mathcal{O}((b+c)^3 \cdot \|R\|^3 \cdot N^9)$.*

Proof: We apply Algorithm 7 to compose the parametric DBM $k \cdot \Lambda + \sigma(R^b)$ with $\sigma(R^c)$, which requires time $\mathcal{O}(\mu^3 \cdot N^6)$, cf. Proposition 12. By an argument similar to the one used in the proof of Proposition 13, we obtain $\mu \leq (b + c + 1) \cdot N \cdot \|R\|$. The result of MAXPERIODIC is the κ bound from Lemma 11, which can be established during the computation of the min-sets using Algorithm 7. Hence the result follows. □

Finally, we prove the asymptotic complexity on the running of Algorithm 5 for a difference bounds relation R in terms of its prefix, period, the number of variables used to define R, and the sum of absolute values of coefficients of R.

Theorem 8. *Let R be a difference bounds relation with prefix B and period C. Then, Algorithm 5 computes the transitive closure of R in at most $\mathcal{O}((B + C)^8 \cdot \|R\|^3 \cdot N^9)$ time.*

Proof: By Theorem 2, Algorithm 5 takes at most $\mathcal{O}((B + C)^2)$ iterations of the main loop and in each iteration and moreover, the algorithm considers a prefix and period candidates b and c such that both b and c are bounded by $\mathcal{O}((B+C)^2)$. By Proposition 13, Procedure MAXCONSISTENT runs in time at most $\mathcal{O}((b + c)^3 \cdot \|R\|^3 \cdot N^9)$. Combining this bound with the bound on b and c, if follows that MAXCONSISTENT runs in time at most $\mathcal{O}((B + C)^6 \cdot \|R\|^3 \cdot N^9)$. We obtain the same bound on running time of MAXPERIODIC, by Proposition 14. The test on line 8 can be performed in $\mathcal{O}(N^2)$ time, by Proposition 3. The greatest power of a relation that is computed by the algorithm is R^{b+2c}. Since the composition of difference bounds relations can be computed in $\mathcal{O}(N^3)$ time, if follows that these computations are performed in $\mathcal{O}((B + C) \cdot N^3)$ time. Since the algorithm takes at most $\mathcal{O}((B + C)^2)$ iterations, we finally infer that the total running time of Algorithm 5 is bounded by $\mathcal{O}((B + C)^8 \cdot \|R\|^3 \cdot N^9)$. □

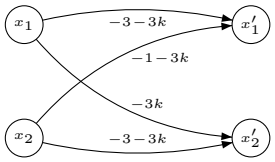
Running Example.. We demonstrate the main steps of Algorithm 5 applied to the difference bounds relation $R \Leftrightarrow x_1 - x_1' \leq 1 \wedge x_1 - x_2' \leq -1 \wedge x_2 - x_1' \leq -2 \wedge x_2 - x_2' \leq 2$. The first valid guess for $(b, c) = (2, 2)$, for which the test on line 9 succeeds, leads to the candidate rate Λ (Figure 9). The MAXCONSISTENT procedure first computes the parametric DBM corresponding to $\rho(k \cdot \Lambda + \sigma(R^b))$, shown in Figure 10a. The DBM

$$M_{R^b} = M_{R^c} = \begin{pmatrix} 0 & \infty & -3 & 0 \\ \infty & 0 & -1 & -3 \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} M_{R^{b+c}} = \begin{pmatrix} 0 & \infty & -6 & 3 \\ \infty & 0 & -4 & -6 \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} \Lambda = \begin{pmatrix} 0 & \infty & -3 & -3 \\ \infty & 0 & -3 & -3 \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix}$$

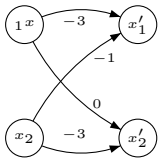
Figure 9: Candidate rate Λ for $(b, c) = (2, 2)$.

$$\begin{pmatrix} 0 & \infty & (-3-3k) & -3k \\ \infty & 0 & (-1-3k) & (-3-3k) \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix}$$

(a) $k \cdot \Lambda + M_{R^b}$



(b) $\mathcal{G}_{\sigma(k\cdot\Lambda+M_{R^b})}$



(c) $\mathcal{G}_{R^c}$

Figure 10: Left-hand side of the MAXPERIODIC equivalence test

is already closed and thus, application of Algorithm 5 doesn't change its entries. Next, Lemma 9 is applied to compute the value $K = \infty$ that MAXCONSISTENT returns.

The MAXPERIODIC procedure checks that

$$\rho(k \cdot \Lambda + \sigma(R^b)) \circ R^c \Leftrightarrow \rho((k+1) \cdot \Lambda + \sigma(R^b))$$

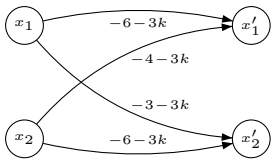
for all $k \geq 0$. The parametric DBM $\rho((k+1) \cdot \Lambda + \sigma(R^b))$ representing the right-hand side of the equivalence is shown in Figure 11. The left-hand side is equivalent to the composition of $\sigma(k \cdot \Lambda + M_{R^b})$ (Figures 10a and 10b) with R^c (Figure 10c). This amounts to the computation of shortest paths between extremal vertices of the graph in Figure 12 which results in a graph identical to the one in Figure 11. Since this graph represents the right-hand side, the above equivalence holds for all $k \geq 0$ and thus, MAXPERIODIC returns $L = \infty$.

Then, a test on line 12 succeeds and Algorithm 5 returns the transitive closure:

$$\begin{aligned} &R^+ \Leftrightarrow \textstyle\bigvee_{i=1}^{b-1} R^i \ \vee \ \exists k \geq 0 \,.\, \bigvee_{i=0}^{c-1} \pi(k \cdot \Lambda + \sigma(R^2)) \circ R^i \Leftrightarrow \\ &(x_1 - x'_1 \leq 1 \wedge x_1 - x'_2 \leq -1 \wedge x_2 - x'_1 \leq -2 \wedge x_2 - x'_2 \leq -2) \vee \exists k \geq 0 \,. \\ &(x_1 - x'_1 \leq -3k-3 \wedge x_1 - x'_2 \leq -3k \wedge x_2 - x'_1 \leq -3k-1 \wedge x_2 - x'_2 \leq -3k-3) \vee \\ &(x_1 - x'_1 \leq -3k-2 \wedge x_1 - x'_2 \leq -3k-4 \wedge x_2 - x'_1 \leq -3k-5 \wedge x_2 - x'_2 \leq -3k-2) \end{aligned}$$

$$\begin{pmatrix} 0 & \infty & (-6-3k) & (-3-3k) \\ \infty & 0 & (-4-3k) & (-6-3k) \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix}$$

(a) $(k+1) \cdot \Lambda + M_{R^b}$



(b) $\mathcal{G}_{\sigma((k+1)\cdot\Lambda+M_{R^b})}$

Figure 11: Right-hand side of the MAXPERIODIC equivalence test

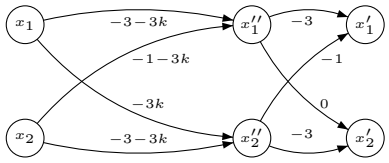


Figure 12: Computing parametric composition $\sigma(k \cdot \Lambda + M_{R^b}) \circ R^c$.

After the elimination of the existential quantifier, we obtain:

$$\begin{aligned} R^+ \quad \Leftrightarrow \quad & (x_1 - x_1' \leq 1 \wedge x_1 - x_2' \leq -1 \wedge x_2 - x_1' \leq -2 \wedge x_2 - x_2' \leq -2) \vee \\ & (x_1 - x_1' \leq -3 \wedge x_1 - x_2' \leq 0 \wedge x_2 - x_1' \leq -1 \wedge x_2 - x_2' \leq -3) \vee \\ & (x_1 - x_1' \leq -2 \wedge x_1 - x_2' \leq -4 \wedge x_2 - x_1' \leq -5 \wedge x_2 - x_2' \leq -2) \end{aligned}$$

7.5. Octagonal Relations

Recall from Section 6.2 that an octagonal relation $R \subseteq \mathbb{Z}^N \times \mathbb{Z}^N$ defined by a formula $R(\mathbf{x}, \mathbf{x}')$ can be represented as a difference bounds relation $\overline{R}(\mathbf{y}, \mathbf{y}')$ defined over the dual set of variables with the convention that y_{2i-1} stands for x_i and y_{2i} for $-x_i$. Then, an octagonal relation R can be represented by a difference bounds matrix $M_{\overline{R}}$. Similarly, for each DBM M, there is a corresponding octagonal relation Ω_M. Furthermore, octagonal relations can be represented canonically by DBMs $M^t_{\overline{R}}$.

We start by defining the mappings between octagonal relations and their matrix encodings required by Definition 4. Given a consistent octagonal relation $R(\mathbf{x}, \mathbf{x}')$ let $\sigma(R) = M^t_{\overline{R}}$. Dually, for any coherent DBM $M \in \mathbb{Z}_\infty^{4N \times 4N}$, let $\rho(M) = \Omega_M$. Clearly, $\rho(\sigma(R)) \Leftrightarrow R$, as required by Definition 4. In order to define the mapping π, we first define the class of *parametric* octagonal relations.

Definition 22. *A formula $\phi(\mathbf{x}, k)$ is a* parametric octagonal constraint *if it is equivalent to a finite conjunction of terms of the form $x_i - x_j \leq u_{ij}$, $x_i + x_j \leq v_{ij}$ or $x_i + x_j \geq t_{ij}$, where u_{ij}, v_{ij} and t_{ij} are univariate linear terms in k, for all $1 \leq i, j \leq N$.*

A parametric octagon $\phi(\mathbf{x}, k)$ is represented by a matrix $M_{\overline{\phi}}[k] \in \mathbb{Z}[k]^{2N \times 2N}_\infty$ of univariate linear terms. Vice versa, a matrix $M[k] \in \mathbb{Z}[k]^{2N \times 2N}_\infty$ encodes a parametric octagonal relation, denoted as $\Omega_M(k)$. With these considerations, we define $\pi(M[k]) = \Omega_M(k)$ to be the parametric counterpart of the ρ function from Definition 4.

7.6. Proving Periodicity

In order to prove that the class $\mathcal{R}_{oct}$ of octagonal relations is periodic, we need to prove that the sequence $\{\sigma(R^m)\}_{m=0}^\infty$ is periodic, for an arbitrary relation $R \in \mathcal{R}_{oct}$. It is sufficient to consider only the case where R is $*$-consistent i.e., $\sigma(R^m) = M^t_{\overline{R^m}}$, for all $m \geq 0$. We rely in the following on Theorem 4 which gives a method to compute $M^t_{\overline{R^m}}$, the tightly closed DBM representation of R^m, from $M^*_{\overline{R}^m}$, the closed DBM representation of $\overline{R}^m$.

We have previously shown, in Section 7.2, that difference bounds relations are periodic. In particular, this means that the sequence $\{M^*_{\overline{R}^m}\}_{m=0}^{\infty}$, corresponding to the iteration of the difference bounds relation $\overline{R}$, is periodic. To prove that the sequence $\{M^t_{\overline{R}^m}\}_{m=0}^{\infty}$ is also periodic, it is sufficient to show that (i) the minimum and (ii) the sum of two periodic sequences are periodic, and also that (iii) the integer half of a periodic sequence is also periodic.

Lemma 12. *Let $\{s_m\}_{m=0}^{\infty}$ and $\{t_m\}_{m=0}^{\infty}$ be two periodic sequences. Then the sequences $\{\min(s_m, t_m)\}_{m=0}^{\infty}$, $\{s_m + t_m\}_{m=0}^{\infty}$ and $\left\{\left\lfloor \frac{s_m}{2} \right\rfloor\right\}_{m=0}^{\infty}$ are periodic as well. Let b_s (c_s) be the prefix (period) of $\{s_m\}_{m=0}^{\infty}$, let b_t (c_t) be the prefix (period) of $\{t_m\}_{m=0}^{\infty}$, and let define $b = \max(b_s, b_t)$, $c = \mathrm{lcm}(c_s, c_t)$, and $b_m = b + \max_{i=0}^{c-1} K_i c$, where*

$$K_i = \begin{cases} \left\lceil \frac{s_{b+i} - t_{b+i}}{\lambda_i^{(t)} - \lambda_i^{(s)}} \right\rceil & \text{if } \lambda_i^{(s)} < \lambda_i^{(t)} \text{ and } t_{b+i} < s_{b+i} \\ \left\lceil \frac{t_{b+i} - s_{b+i}}{\lambda_i^{(s)} - \lambda_i^{(t)}} \right\rceil & \text{if } \lambda_i^{(t)} < \lambda_i^{(s)} \text{ and } s_{b+i} < t_{b+i} \\ 0 & \text{otherwise} \end{cases}$$

for each $i = 0, \ldots, c-1$ and where $\lambda_0^{(s)}, ..., \lambda_{c-1}^{(s)}$ ($\lambda_0^{(t)}, ..., \lambda_{c-1}^{(t)}$) are rates of $\{s_m\}_{m=0}^{\infty}$ ($\{t_m\}_{m=0}^{\infty}$) with respect to the common prefix b and period c. Then, the prefix and the period of the above sequences are:

	prefix	*period*
$\{s_m + t_m\}_{m=0}^{\infty}$	b	c
$\left\{\left\lfloor \frac{s_m}{2} \right\rfloor\right\}_{m=0}^{\infty}$	b	$2c$
$\{\min(s_m, t_m)\}_{m=0}^{\infty}$	b_m	c

Proof: We can show that the sum sequence $\{s_m + t_m\}_{m=0}^{\infty}$ is periodic as well, with prefix b, period c and rates $\lambda_0^{(s)} + \lambda_0^{(t)}, ..., \lambda_{c-1}^{(s)} + \lambda_{c-1}^{(t)}$. In fact, for every $k \geq 0$ and $i = 0, ..., c-1$ we have successively:

$$\begin{aligned} (s+t)_{b+(k+1)c+i} &= s_{b+(k+1)c+i} + t_{b+(k+1)c+i} && (8) \\ &= \lambda_i^{(s)} + s_{b+kc+i} + \lambda_i^{(t)} + t_{b+kc+i} && (9) \\ &= \lambda_i^{(s)} + \lambda_i^{(t)} + s_{b+kc+i} + t_{b+kc+i} && (10) \\ &= (\lambda_i^{(s)} + \lambda_i^{(t)}) + (s+t)_{b+kc+i} && (11) \end{aligned}$$

For the min sequence $\{\min(s_m, t_m)\}_{m=0}^{\infty}$, it can be shown that, for each $i = 0, ..., c-1$ precisely one of the following assertions hold:

1. ($\lambda_i^{(s)} < \lambda_i^{(t)}$ or $\lambda_i^{(s)} = \lambda_i^{(t)}$ and $s_{b+i} < t_{b+i}$) and $\forall k \geq 0$. $s_{b+K_i c+kc+i} \leq t_{b+K_i c+kc+i}$
2. ($\lambda_i^{(t)} < \lambda_i^{(s)}$ or $\lambda_i^{(s)} = \lambda_i^{(t)}$ and $t_{b+i} < s_{b+i}$) and $\forall k \geq 0$. $t_{b+K_i c+kc+i} \leq s_{b+K_i c+kc+i}$

Intuitively, starting from the position $b + K_i c$, on every period c, the minimum amongst the two sequences is always defined by the same sequence i.e., the one having

the minimal rate on index i, or if the rates are equal, the one having the smaller starting value.

We can show now that the min sequence $\{\min(s_m, t_m)\}_{m=0}^{\infty}$ is periodic starting at $b_m = b + \max_{i=0}^{c-1} K_i c$, with period c and rates $\min(\lambda_0^{(s)}, \lambda_0^{(t)}), ..., \min(\lambda_{c-1}^{(s)}, \lambda_{c-1}^{(t)})$. That is, we have successively, for every $k \geq 0$ and $i = 0, ..., c-1$, and whenever i satisfies the condition (1) above (the case when i satisfies the condition (2) being similar):

$$\begin{aligned}\min(s_{b_m+(k+1)c+i}, t_{b_m+(k+1)c+i}) &= s_{b_m+(k+1)c+i} \\ &= \lambda_i^{(s)} + s_{b_m+kc+i} \\ &= \min(\lambda_i^{(s)}, \lambda_i^{(t)}) + \min(s_{b_m+kc+i}, t_{b_m+kc+i})\end{aligned}$$

For the sequence $\left\{\left\lfloor \frac{s_m}{2} \right\rfloor\right\}_{m=0}^{\infty}$, assume that the sequence $\{s_m\}_{m=0}^{\infty}$ is periodic with prefix b, period c and rates $\lambda_0, ..., \lambda_{c-1}$. It can be easily shown that the sequence $\lfloor \frac{s_m}{2} \rfloor$ is periodic as well with prefix b, period $2c$, and rates $\lambda_0, ..., \lambda_{c-1}, \lambda_0, ..., \lambda_{c-1}$.

We have successively for any $k \geq 0$, and for any $i = 0, ..., c-1$:

$$\left\lfloor \frac{s_{b+(k+1)2c+i}}{2} \right\rfloor = \left\lfloor \frac{2\lambda_i + s_{b+k\cdot 2c+i}}{2} \right\rfloor = \lambda_i + \left\lfloor \frac{s_{b+k\cdot 2c+i}}{2} \right\rfloor$$

Similarly, for any $k \geq 0$ and for any $i = 0, ..., c-1$, we have:

$$\left\lfloor \frac{s_{(b+k+1)2c+c+i}}{2} \right\rfloor = \left\lfloor \frac{2\lambda_i + s_{b+k\cdot 2c+c+i}}{2} \right\rfloor = \lambda_i + \left\lfloor \frac{s_{b+k\cdot 2c+c+i}}{2} \right\rfloor$$

□

The theorem below is an immediate consequence of Theorem 4 and Lemma 12.

Theorem 9. *The class of octagonal relations is periodic.*

Corollary 2. *If $R \in \mathcal{R}_{oct}$ is an octagonal relation, the rate Λ of the periodic sequence $\{\sigma(R^i)\}_{i=0}^{\infty}$ is tightly closed.*

Proof: On one hand, $\sigma(R^i) = M_{R^i}^t$ is tightly closed, by definition. By Theorem 9, there exist $b \geq 0$, $c > 0$ and $\Lambda \in \mathbb{Z}_{\infty}^{4N \times 4N}$, for all $n \geq 0$:

$$\sigma(R^{nc+b}) = n \cdot \Lambda + \sigma(R^b)$$

The first two points of Definition 16 are immediate. The closure (point 3 of Definition 16) is by Corollary 1. We are left with proving the last point, namely that for all $1 \leq i, j \leq 4N$:

$$\Lambda_{ij} \leq \lfloor \frac{\Lambda_{i\bar{i}}}{2} \rfloor + \lfloor \frac{\Lambda_{\bar{j}j}}{2} \rfloor \tag{12}$$

Since $\sigma(R^{nc+b})$ is tightly closed, for all $n \geq 0$, we have, for all $1 \leq i, j \leq 4N$:

$$\begin{aligned} n \cdot \Lambda_{ij} + \sigma(R^b)_{ij} &\leq \lfloor \tfrac{n\cdot\Lambda_{i\bar{i}} + \sigma(R^b)_{i\bar{i}}}{2} \rfloor + \lfloor \tfrac{n\cdot\Lambda_{\bar{j}j} + \sigma(R^b)_{\bar{j}j}}{2} \rfloor \\ &\leq n \cdot (\lfloor \tfrac{\Lambda_{i\bar{i}}}{2} \rfloor + \lfloor \tfrac{\Lambda_{\bar{j}j}}{2} \rfloor) + \lfloor \tfrac{\sigma(R^b)_{i\bar{i}}}{2} \rfloor + \lfloor \tfrac{\sigma(R^b)_{\bar{j}j}}{2} \rfloor + 2 \end{aligned}$$

The last inequality holds because, for all $x, y \in \mathbb{Z}$ and $n \geq 0$:

- $\lfloor \frac{x+y}{2} \rfloor \leq \lfloor \frac{x}{2} \rfloor + \lfloor \frac{y}{2} \rfloor + 1$,
- $\lfloor \frac{n \cdot x}{2} \rfloor = n \cdot \lfloor \frac{x}{2} \rfloor$ if x is even,

and $\Lambda_{i\bar{i}}$ and $\Lambda_{\bar{j}j}$ are both even. We calculate:

$$n \cdot (\Lambda_{ij} - \lfloor \frac{\Lambda_{i\bar{i}}}{2} \rfloor - \lfloor \frac{\Lambda_{\bar{j}j}}{2} \rfloor) \leq \lfloor \frac{\sigma(R^b)_{i\bar{i}}}{2} \rfloor + \lfloor \frac{\sigma(R^b)_{\bar{j}j}}{2} \rfloor - \sigma(R^b)_{ij} + 2, \ \forall n \geq 0$$

Then the condition (12) follows. □

7.7. Checking $*$-consistency and Periodicity

Similar to the case of difference bounds relations, in this section we give efficient ways to implement the MAXCONSISTENT and MAXPERIODIC procedures from Algorithm 5. In the rest of this section, *a univariate linear half-term* is a term of the form $\lfloor \frac{\alpha \cdot k + \beta}{2} \rfloor$, where the mapping $x \mapsto \lfloor \frac{x}{2} \rfloor$ denotes the integer division by two.

Unlike the case of octagonal constraints with constants coefficients, the matrices representing parametric octagons do not have a tightly closed canonical form. To overcome this problem, one can use Algorithm 7 and Theorem 3 to define the tight closure of a parametric octagonal matrix as a matrix whose entries are either ∞ or terms of the form $\min\{t_i(k)\}_{i=1}^m$, where $t_i(k)$ are either univariate linear terms or sums of half-terms.

MAXCONSISTENT.. Given an octagonal relation R, integers $b \geq 0, c > 0$ such that R^{b+2c} is consistent, and a matrix $\Lambda \in \mathbb{Z}_\infty^{4N \times 4N}$, let us denote $M_{R,b,\Lambda} = k \cdot \Lambda + \sigma(R^b) \in \mathbb{Z}[k]_\infty^{4N \times 4N}$. Similarly as in the difference bounds case, we have:

$$\begin{array}{rcl} \text{MAXCONSISTENT}(R, b, \Lambda) & = & \sup\{n \in \mathbb{N} \mid M_{R,b,\Lambda}[n] \text{ is octagonal-consistent}\} \\ & = & \inf\{n \in \mathbb{N} \mid M_{R,b,\Lambda}[n] \text{ is octagonal-inconsistent}\} - 1 \end{array}$$

According to Theorem 3, $M_{R,b,\Lambda}[n]$ is octagonal-inconsistent for some valuation $n \in \mathbb{N}$ of k, if either

(i) $M[n]_{R,b,\Lambda}$ is inconsistent, or

(ii) $\lfloor \frac{(M[n]^*_{R,b,\Lambda})_{i\bar{i}}}{2} \rfloor + \lfloor \frac{(M[n]^*_{R,b,\Lambda})_{\bar{i}i}}{2} \rfloor < 0$ for some $1 \leq i \leq 4N$.

Let $\mathcal{M} = \text{PARAMETRICFW}(M_{R,b,\Lambda})$, as returned by Algorithm 7. Checking for the case (i) can be done in a similar way as for difference bounds constraints (Section 7.2). The condition of case (ii) is equivalent to the following:

$$\min [\![\{ \lfloor \frac{t}{2} \rfloor + \lfloor \frac{u}{2} \rfloor \mid t \in \mathcal{M}_{i\bar{i}}, u \in \mathcal{M}_{\bar{i}i} \}]\!]_{\geq 0} < 0, \text{ for some } 1 \leq i \leq 4N \qquad (13)$$

The following lemma shows that a set of sums of univariate half-terms is semantically equivalent to the union of two sets of univariate linear terms.

Lemma 13. *Let $T = \{t_i + \sum_{j=1}^{m_i} \lfloor \frac{u_{ij}}{2} \rfloor\}_{i=1}^n$, where $t_i = \alpha_i \cdot k + \beta_i, u_{ij} = \gamma_{ij} \cdot k + \delta_{ij}$ are univariate linear terms. Then there exist two sets $\mathcal{L}, \mathcal{U}$ of univariate linear terms such that for all $k \geq 0$:*

$$\begin{array}{rcl} \min\{\mathcal{L}(k)\} & = & \min\{T(2k)\} \\ \min\{\mathcal{U}(k)\} & = & \min\{T(2k+1)\} \end{array}$$

Moreover $\|\mathcal{L}\| \leq \|T\|$ and $\|\mathcal{U}\| \leq \|T\|$.

Proof: For a univariate linear term $\gamma \cdot k + \delta \in \mathbb{Z}[k]$, we have:

$$[\![\lfloor \frac{\gamma \cdot k + \delta}{2} \rfloor]\!]_{\geq 0} = [\![\gamma \cdot k + \lfloor \frac{\delta}{2} \rfloor]\!]_{\geq 0} \cup [\![\gamma \cdot k + \lfloor \frac{\gamma + \delta}{2} \rfloor]\!]_{\geq 0}.$$

Let

$$\begin{array}{rcl} \mathcal{L} & = & \{(2\alpha_i + \sum_{j=1}^{m_i} \gamma_{ij}) \cdot k + \beta_i + \sum_{j=1}^{m_i} \lfloor \frac{\delta_{ij}}{2} \rfloor\}_{i=1}^{n}, \text{ and} \\ \mathcal{U} & = & \{(2\alpha_i + \sum_{j=1}^{m_i} \gamma_{ij}) \cdot k + \alpha_i + \beta_i + \sum_{j=1}^{m_i} \lfloor \frac{\gamma_{ij} + \delta_{ij}}{2} \rfloor\}_{i=1}^{n}. \end{array}$$

Clearly $[\![T]\!]_{\geq 0} = [\![\mathcal{L}]\!]_{\geq 0} \cup [\![\mathcal{U}]\!]_{\geq 0}$, $\min\{\mathcal{L}(k)\} = \min\{T(2k)\}$, and $\min\{\mathcal{U}(k)\} = \min\{T(2k+1)\}$ for all $k \geq 0$. It is easy to see that $\|\mathcal{L}\| \leq \|T\|$ and $\|\mathcal{U}\| \leq \|T\|$. □

Let us denote $\mathcal{T}_i = \{\lfloor \frac{t}{2} \rfloor + \lfloor \frac{u}{2} \rfloor \mid t \in \mathcal{M}_{i\bar{i}}, u \in \mathcal{M}_{\bar{i}i}\}$. Then, the condition (13) can be rewritten as

$$\min [\![\mathcal{T}_i]\!]_{\geq 0} < 0, \text{ for some } 1 \leq i \leq 4N.$$

By Lemma 13, for each $i = 1, \ldots, 4N$ there exist sets of univariate linear terms $\mathcal{L}_i$ and $\mathcal{U}_i$ such that $\min\{\mathcal{L}_i(k)\} = \min\{\mathcal{T}_i(2k)\}$ and $\min\{\mathcal{U}_i(k)\} = \min\{\mathcal{T}_i(2k+1)\}$ for all $k \geq 0$. Therefore, for all $1 \leq i \leq 4N$, we have

$$(\min [\![\mathcal{T}_i]\!]_{\geq 0} < 0) \Leftrightarrow (\min [\![\mathcal{L}_i]\!]_{\geq 0} < 0) \vee (\min [\![\mathcal{U}_i]\!]_{\geq 0} < 0).$$

With these considerations, the MAXCONSISTENT procedure can be implemented as follows. We define

$$\begin{array}{rcl} K_{db} & = & \min\{\Gamma(\mathcal{M}_{ii})\}_{i=1}^{4N}, \\ K_{\mathcal{L}} & = & \min\{\Gamma(\mathcal{L}_i)\}_{i=1}^{4N}, \\ K_{\mathcal{U}} & = & \min\{\Gamma(\mathcal{U}_i)\}_{i=1}^{4N} \end{array}$$

where given a set of univariate linear terms $\mathcal{T}$, $\Gamma(\mathcal{T})$ is a constant defined in the Lemma 9. Note that K_{db} is the minimal integer $n \in \mathbb{N}$ such that $M[n]_{R,b,\Lambda}$ is inconsistent. By Lemma 13, $\min\{2 \cdot K_{\mathcal{L}}, 2 \cdot K_{\mathcal{U}} - 1\}$ is the minimal integer for which (13) holds, or equivalently, the minimal integer $n \in \mathbb{N}$ such that $\lfloor \frac{(M[n]^*_{R,b,\Lambda})_{i\bar{i}}}{2} \rfloor + \lfloor \frac{(M[n]^*_{R,b,\Lambda})_{\bar{i}i}}{2} \rfloor < 0$, for some $1 \leq i \leq 4N$. Consequently, the MAXCONSISTENT procedure returns:

$$\text{MAXCONSISTENT}(R, b, \Lambda) \quad = \quad \min\{K_{db}, 2 \cdot K_{\mathcal{L}}, 2 \cdot K_{\mathcal{U}} - 1\} - 1.$$

Proposition 15. *For an octagonal relation R, an integer $b \geq 0$ such that R^b is consistent and a matrix $\Lambda \in \mathbb{Z}_{\infty}^{4N \times 4N}$,* MAXCONSISTENT *runs in time at most $\mathcal{O}((b+c)^3 \cdot \|R\|^3 \cdot N^9)$.*

Proof: Computing $\mathcal{M}$ requires one application of Algorithm 7. By Proposition 12, the call to Algorithm 7 requires time at most $\mathcal{O}(\mu^3 \cdot N^6)$, where:

$$\mu = \min(\max_{1 \leq i,j \leq 2N} \{|\Lambda_{ij}|\}, \max_{1 \leq i,j \leq 2N} \{|(\sigma(R^b))_{ij}|\})$$

Moreover, the size of $\mathcal{M}_{ij}$ is bounded by $8N \cdot \mu$, by Proposition 12. By an argument similar to the one in the proof of Proposition 12, one infers that $\mu \leq (b+c+1) \cdot 2N \cdot \|R\|$. Consequently, $\mathcal{M}$ takes at most $\mathcal{O}((b+c)^3 \cdot \|R\|^3 \cdot N^9)$ time and $\|\mathcal{M}_{ij}\|$ is bounded by $\mathcal{O}((b+c) \cdot \|R\| \cdot N^2)$.

Hence, computing $\mathcal{L}_i$ and $\mathcal{U}_i$ can be done in time at most $\mathcal{O}(N^2 \cdot (b+c) \cdot \|R\|)$. By Lemma 13, $\|\mathcal{L}_i\| \leq \|\mathcal{M}_{ij}\|$, and $\|\mathcal{U}_i\| \leq \|\mathcal{M}_{ij}\|$ and consequently, $K_{db}, K_{\mathcal{L}}$, an $K_{\mathcal{L}}$ can be computed in $\mathcal{O}((b+c) \cdot \|R\| \cdot N^3)$ time for each $1 \leq i \leq 4N$. Hence, MAXCONSISTENT procedure runs in time at most $\mathcal{O}((b+c)^3 \cdot \|R\|^3 \cdot N^9)$. □

MAXPERIODIC.. Given an octagonal relation R, two integers $b \geq 0$ and $c > 0$, such that R^b is consistent, a constant $K \in \mathbb{N}_\infty$, and a matrix $\Lambda \in \mathbb{Z}_\infty^{4N \times 4N}$, the procedure MAXPERIODIC(R, b, Λ, c, K) returns the maximal positive integer $n \leq K$ such that the equivalence $\rho(\ell \cdot \Lambda + \sigma(R^b)) \circ R^c \Leftrightarrow \rho((\ell+1) \cdot \Lambda + \sigma(R^b))$ holds for all $0 \leq \ell < n$, or ∞ if the above holds for all positive ℓ.

The left-hand side of the equivalence can be encoded by a matrix M_1 of terms of the form $\min\{t_i\}_{i=1}^m$, where t_i are either univariate linear terms or sums of univariate half-terms, and can be computed by Algorithm 7. We show that the DBM M_2 that encodes the right-hand side relation is tightly closed, for all valuation of k, meaning that the right-hand side can be simply represented by a parametric DBM, under the octagonal interpretation.

Lemma 14. *Let $R \in \mathcal{R}_{oct}$ be an octagonal relation, and Λ be the rate of the periodic sequence $\{\sigma(R^i)\}_{i=0}^\infty$. Then, for all $b, n \geq 0$, the DBM $(n+1) \cdot \Lambda + \sigma(R^b)$ is tightly closed.*

Proof: As a direct consequence of the fact that $\sigma(R^b)$ is tightly closed, by definition, and that Λ is tightly closed, by Corollary 2. □

Two octagonal relations are equivalent whenever their tightly closed DBM encodings are equal (Proposition 6). Hence we need to check for equality (inside some interval) between min-sets of univariate linear terms and sums of half-terms (the left-hand side matrix M_1) and univariate linear terms (the right-hand side matrix M_2). Here again Lemma 13 comes to rescue. We compute M_1 using Algorithm 7 and Theorem 4. Using Lemma 13, we split M_1 to $M_{1,L}$ and $M_{1,U}$, where $M_{1,L}, M_{1,U}$ are matrices with sets of univariate terms as entries such that $\min\{(M_{1,L})_{ij}(k)\} = \min\{M_{ij}(2k)\}$ and $\min\{(M_{1,U})_{ij}(k)\} = \min\{M_{ij}(2k+1)\}$ for all $k \geq 0$. Similarly, we split M_2 to $M_{2,L}$ and $M_{2,U}$. Then, we apply Lemma 11 to compute the upper bound P_L (P_U) of the interval in which $M_{1,L}$ and $M_{2,L}$ ($M_{1,U}$ and $M_{2,U}$) are equal. Finally, the upper bound of the interval in which M_1 and M_2 are equal is computed as $\min\{2 \cdot P_L, 2 \cdot P_U - 1\} + 1$.

Proposition 16. *For a difference bounds relation R, and integers $b \geq 0, c > 0$ such that R^{b+c} is consistent and a matrix $\Lambda \in \mathbb{Z}_\infty^{2N \times 2N}$,* MAXPERIODIC *runs in time at most $\mathcal{O}((b+c)^3 \cdot \|R\|^3 \cdot N^9)$.*

Proof: By an argument similar to the one used in the proof of Proposition 15, Algorithm 7 computes the matrix $\mathcal{M}$ of sets representing the parametric composition of $k \cdot \Lambda + \sigma(R^b)$ with $\sigma(R^c)$ in time $\mathcal{O}((b+c)^3 \cdot \|R\|^3 \cdot N^9)$. Moreover the size of each entry $\mathcal{M}_{ij}$ is bounded by $\mathcal{O}((b+c) \cdot \|R\| \cdot N^2)$. Computing the minimal bound from Lemma 11 requires then $\mathcal{O}((b+c) \cdot \|R\| \cdot N^4)$ time. Hence the result. □

Finally, we prove the asymptotic complexity on the running of Algorithm 5 for an octagonal relation R in terms of its prefix, period, the number of variables used to define R, and the sum of absolute values of coefficients of R.

Theorem 10. *Let R be an octagonal relation with prefix B and period C. Then, Algorithm 5 computes the transitive closure of R in at most $\mathcal{O}((B+C)^8 \cdot \|R\|^3 \cdot N^9)$ time.*

Proof: The bounds on the running time of procedures MAXCONSISTENT (Propositions 15) and MAXPERIODIC (Proposition 16) for octagonal relations are same as for difference bounds relations (Propositions 13 and 14, respectively). Similarly, relational composition of octagons has the same asymptotic bound $\mathcal{O}(N^3)$ as difference bounds constraints. Hence, we obtain the same bound as in Theorem 8. □

Running Example. Consider the octagonal relation $R(x_1, x_2, x_1', x_2') \Leftrightarrow x_1 + x_2 \leq 5 \wedge x_1' - x_1 \leq -2 \wedge x_2' - x_2 \leq -3 \wedge x_2' - x_1' \leq 1$. The check at line 9 of Algorithm 5 succeeds for $(b, c) = (1, 1)$. In order to compute MAXPERIODIC, one needs to compose parametric difference bound matrices, similarly as in the case of difference bound relations. Moreover, the tightening step must be performed. Figure 13 shows the parametric matrix M representing the left-hand side of the equivalence checked by MAXPERIODIC. The matrix is closed, but not tightly closed. We illustrate the tightening for the constraint $y_3'' - y_4'$, which depends on constraints $y_3'' - y_4''$ and $y_4'' - y_4''$ and is thus computed as

$$\min\{-6k - 3, \lfloor \frac{-6k}{2} \rfloor + \lfloor \frac{-6k - 6}{2} \rfloor\}$$

By Lemma 13, we obtain

$$\begin{array}{lcllcl} [\![-6k-3]\!]_{\geq 0} & = & L_1 \cup U_1 & = & [\![-12k-3]\!]_{\geq 0} & \cup \; [\![-12k-9]\!]_{\geq 0} \\ [\![\lfloor \frac{-6k}{2} \rfloor]\!]_{\geq 0} & = & L_2 \cup U_2 & = & [\![-6k]\!]_{\geq 0} & \cup \; [\![-6k-3]\!]_{\geq 0} \\ [\![\lfloor \frac{-6k-6}{2} \rfloor]\!]_{\geq 0} & = & L_3 \cup U_3 & = & [\![-6k-3]\!]_{\geq 0} & \cup \; [\![-6k-6]\!]_{\geq 0} \end{array}$$

The tightening step splits M into M_L and M_U. In M_L, the tightened constraint $y_3'' - y_4'$ is computed as

$$\min\{-12k - 3, (-6k) + (-6k - 3)\} = -12k - 3$$

and similarly in M_U, the tightened constraint $y_3'' - y_4'$ is computed as

$$\min\{-12k - 9, (-6k - 3) + (-6k - 6)\} = -12k - 9$$

Further checks are similar as in the difference bounds case. Then, Algorithm 5 returns the following result:

$$\begin{array}{rcl} R^+ & \Leftrightarrow & \bigvee_{i=1}^{b-1} R^i \;\vee\; \exists k \geq 0 \,.\, \bigvee_{i=0}^{c-1} \pi(k \cdot \Lambda + \sigma(R)) \circ R^i \\ & \Leftrightarrow & \exists k \geq 0 \,.\, x_2' \leq -3k \wedge x_1' - x_1 \leq -2k - 2 \wedge x_2' - x_1' \leq 1 \wedge x_2' - x_1 \leq -3k - 1 \wedge \\ & & x_2' - x_2 \leq -3k - 3 \wedge x_1 + x_2 \leq 5 \wedge x_1 + x_2' \leq -3k + 2 \wedge \\ & & x_1' + x_2 \leq -2k + 3 \wedge x_1' + x_2' \leq -5k \wedge x_2' + x_2 \leq -3k + 4 \end{array}$$

After quantifier elimination, we obtain:

$$\begin{array}{rcl} R^+ & \Leftrightarrow & x_2' \leq 0 \wedge x_1' - x_1 \leq -2 \wedge x_2' - x_1 \leq -1 \wedge x_2' - x_1' \leq 1 \wedge \\ & & x_2' - x_2 \leq -3 \wedge x_1 + x_2 \leq 5 \wedge x_1 + x_2' \leq 2 \wedge \\ & & x_1' + x_2 \leq 3 \wedge x_1' + x_2' \leq 0 \wedge x_2 + x_2' \leq 4 \end{array}$$

□

	y_1	y_2	y_3	y_4	y_1''	y_2''	y_3''	y_4''	y_1'	y_2'	y_3'	y_4'
y_1	0	∞	∞	5	∞	∞	∞	$-3k+2$	∞	∞	∞	$-3k-1$
y_2	∞	0	∞	∞	∞	$-2k-2$	∞	$-3k-1$	∞	$-2k-4$	∞	$-3k-4$
y_3	∞	5	0	∞	∞	$-2k+3$	∞	$-3k+4$	∞	$-2k+1$	∞	$-3k+1$
y_4	∞	∞	∞	0	∞	∞	∞	$-3k-3$	∞	∞	∞	$-3k-6$
y_1''	$-2k-2$	∞	∞	$-2k+3$	0	∞	∞	$-5k$	∞	∞	∞	$-5k-3$
y_2''	∞	∞	∞	∞	∞	0	∞	1	∞	-2	∞	-2
y_3''	$-3k-1$	$-3k+2$	$-3k-3$	$-3k+4$	1	$-5k$	0	$-6k$	∞	$-5k-2$	∞	$-6k-3$
y_4''	∞	∞	∞	∞	∞	∞	∞	0	∞	∞	∞	-3
y_1'	$-2k-4$	∞	∞	$-2k+1$	-2	∞	∞	$-5k-2$	0	∞	∞	$-5k-5$
y_2'	∞	∞	∞	∞	∞	∞	∞	∞	∞	0	∞	1
y_3'	$-3k-4$	$-3k-1$	$-3k-6$	$-3k+1$	-2	$-5k-3$	-3	$-6k-3$	1	$-5k-5$	0	$-6k-6$
y_4'	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	0

Figure 13: Left-hand side before tightening

7.8. Finite Monoid Affine Relations

Recall from Section 6.3 that an affine relation $R \in \mathbb{Z}^N \times \mathbb{Z}^N$ is defined by a linear arithmetic constraint of the form $\mathbf{x}' = A\mathbf{x} + \mathbf{b}$, where $A \in \mathbb{Z}^{N\times N}$ is a square matrix, and $\mathbf{b} \in \mathbb{Z}^N$ is a column vector. The relation is said to have the finite monoid property if the set $\{A^0, A^1, \ldots\}$ of matrix powers of A is finite.

It is easy to see that A is finite monoid if and only if there exists $p \geq 0$ and $l > 0$ such that $A^p = A^{p+l}$, i.e. $\mathcal{M}_A = \{A^0, \ldots, A^p, \ldots, A^{p+l-1}\}$. If A has the finite monoid property, it can be shown that the transitive closure of T can be defined in Presburger arithmetic [9, 25]. We achieve the same result below, by showing that the update of a finite monoid affine relations is a periodic relation. As a consequence, the closed form of the update can be computed by Algorithm 5. Since the update relation is $*$-consistent and deterministic, the transitive closure can be computed by applying the following lemma.

Lemma 15. *Let $R(\mathbf{x}, \mathbf{x}') \in \mathcal{R}$ be a $*$-consistent deterministic relation and $\varphi(\mathbf{x})$ be a guard. Then the transitive closure of the relation $R \wedge \varphi$ can be defined as:*

$$(R \wedge \varphi)^+(\mathbf{x}, \mathbf{x}') \Leftrightarrow \exists k > 0 \,.\, \widehat{R}(k, \mathbf{x}, \mathbf{x}') \wedge \forall 0 \leq \ell < k \; \exists \mathbf{y} \,.\, \widehat{R}(\ell, \mathbf{x}, \mathbf{y}) \wedge \varphi(\mathbf{y})$$

where $\widehat{R}$ defines the closed form of R.

Proof: "$\Rightarrow$" Let $\mathbf{v}, \mathbf{v}'$ be a pair of valuations of $\mathbf{x}$ and $\mathbf{x}'$, respectively, such that $\mathbf{v}, \mathbf{v}' \models (R \wedge \varphi)^+$. Then there exists $n > 0$ such that $\mathbf{v}, \mathbf{v}' \models (R \wedge \varphi)^n$. Consequently, there exists a sequence of valuations $\mathbf{v} = \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n = \mathbf{v}'$ such that $\mathbf{v}_i, \mathbf{v}_{i+1} \models R \wedge \varphi$. By Definition 6, we have that $\models \widehat{R}(n, \mathbf{v}_0, \mathbf{v}_n)$ and $\models \widehat{R}(i, \mathbf{v}_0, \mathbf{v}_i) \wedge \varphi(\mathbf{v}_i)$, for all $i = 0, \ldots, n-1$.
"$\Leftarrow$" Let $\mathbf{v}$ and $\mathbf{v}'$ be two valuations such that $\models \widehat{R}(n, \mathbf{v}, \mathbf{v}')$ for some $n > 0$ and for all $i = 0, \ldots, n-1$ we have $\models \widehat{R}(i, \mathbf{v}, \mathbf{v}_i)$ and $\models \varphi(\mathbf{v}_i)$, for some valuation $\mathbf{v}_i$ of $\mathbf{x}$. Since $\widehat{R}(n) \Leftrightarrow R^n$, by Definition 6, there exists a sequence of valuations $\mathbf{v} = \mathbf{v}'_0, \mathbf{v}'_1, \ldots, \mathbf{v}'_n = \mathbf{v}'$ such that $\mathbf{v}'_i, \mathbf{v}'_{i+1} \models R$. By the fact that R was assumed

to be deterministic, we have $\mathbf{v}_i = \mathbf{v'}_i$ for all $i = 0, \ldots, n-1$, hence $\mathbf{v}'_i \models \varphi$, for all $i = 0, \ldots, n-1$. Clearly then $\mathbf{v}, \mathbf{v}' \models (R \wedge \varphi)^+$. □

To compute the transitive closure of an affine relation, it is enough to compute the closed form of its update. This can be computed by Algorithm 5 whenever the update relation is shown to be periodic. In the following, we show that this is indeed the case, when A has the finite monoid property. For simplicity reasons, we will work with the equivalent homogenous form of (4)

$$T_h \Leftrightarrow \mathbf{x'}_h = A_h \times \mathbf{x}_h \ \wedge \ \phi_h(\mathbf{x}_h) \quad \text{where} \quad A_h \Leftrightarrow \left(\begin{array}{c|c} A & \mathbf{b} \\ \hline 0 \ldots 0 & 1 \end{array}\right)$$

where $x_h = \langle x_1, \ldots x_N, x_{N+1} \rangle$ with $x_{N+1} \notin \mathbf{x}$ being a fresh variable and $\phi_h(\mathbf{x}_h) \Leftrightarrow \phi(\mathbf{x}) \wedge x_{N+1} = 1$. The encoding of an affine update $T_u \Leftrightarrow \mathbf{x'}_h = A_h \times \mathbf{x}_h$ is defined as $\sigma(T_u) = A_h \in \mathbb{Z}_\infty^{(N+1)\times(N+1)}$. Dually, for some $M \in \mathbb{Z}[k]_\infty^{(N+1)\times(N+1)}$, we define $\pi(M) \Leftrightarrow \mathbf{x'}_h = M \times \mathbf{x}_h$. With these definitions, we have $\sigma(T_u^k) = {A_h}^k$, for all $k > 0$. The next lemma proves that the class of finite monoid affine updates is periodic.

Lemma 16. *Let $A \in \mathbb{Z}^{N\times N}$ be a finite monoid matrix, $\mathbf{b} \in \mathbb{Z}^N$ be a vector, and let write the finite monoid generated by A as $\mathcal{M}_A = \{A^0, \ldots, A^p, \ldots, A^{p+l-1}\}$, where $p \geq 0$, $l > 0$, and $A^p = A^{p+l}$. Then, the sequence $\{{A_h}^k\}_{k=0}^\infty$ is periodic with prefix p and period l.*

Proof: Let $A \in \mathbb{Z}^{N\times N}$ be a matrix, $\mathbf{b} \in \mathbb{Z}^N$ be a vector, and

$$A_h \Leftrightarrow \left(\begin{array}{c|c} A & \mathbf{b} \\ \hline 0 \ldots 0 & 1 \end{array}\right)$$

Then we have, for all $k \geq 0$:

$$(A_h)^k = \left(\begin{array}{c|c} A^k & \sum_{i=0}^{k-1} A^i \times \mathbf{b} \\ \hline 0 \ldots 0 & 1 \end{array}\right)$$

For $i = N+1,\ 1 \leq j \leq N+1$, $\{({A_h}^k)_{ij}\}_{k=0}^\infty$ is trivially periodic. For $1 \leq i, j \leq N$, $\{({A_h}^k)_{ij}\}_{k=0}^\infty$ is periodic due to the fact that A is finite monoid. It remains to be proven that, for all $1 \leq j \leq N$, the sequence $\{(\sum_{i=0}^{k-1} A^i \times \mathbf{b})_j\}_{k=0}^\infty$ is periodic. Without loss of generality, assume that the monoid of A is $\mathcal{M}_A = \{A^0, A^1, \ldots, A^p, \ldots A^{p+l-1}\}$, where $A^p = A^{p+l}$. Then, for $k \geq p$, we have:

$$\textstyle\sum_{i=0}^{k-1} A^i = \sum_{i=0}^{p-1} A^i + \lfloor \frac{k-p+1}{l} \rfloor \cdot \sum_{i=p}^{p+l-1} A^i + \sum_{i=p}^{p+((k-p+1) \mod l)} A^i.$$

Hence the sequence $\{\sum_{i=0}^{k-1} A^i\}_{k=0}^\infty$ is periodic with prefix p, period l, and rates

$$\Lambda_j = \textstyle\sum_{i=p}^{p+l-1} A^i$$

for all $j = 0, 1, \ldots, l-1$. Consequently, A_h is periodic with the same prefix and period. □

As a direct consequence, we have the following theorem.

Theorem 11. *The class of finite monoid affine updates is periodic. Moreover, the transitive closures of finite monoid affine relations with Presburger definable guards are effectively Presburger definable.*

The implementation of the procedures MAXCONSISTENT and MAXPERIODIC for finite monoid affine relations is rather simple. Since we run Algorithm 5 for $*$-consistent updates of the form $\mathbf{x'}_h = A_h \times \mathbf{x}_h$ only, MAXCONSISTENT needs to return always ∞. The MAXPERIODIC test can be implemented as an equivalence check between two homogeneous linear systems with univariate linear coefficients. More precisely, given a homogeneous transformation $x' = A \times x$, with $A \in \mathbb{Z}^{(N+1)\times(N+1)}$ and a matrix $\Lambda \in \mathbb{Z}^{(N+1)\times(N+1)}$, we are looking for valuations of k that satisfy the following equality

$$(A^b + k \cdot \Lambda) \times A^c = A^b + (k+1) \cdot \Lambda \tag{14}$$

Both $(A^b + k \cdot \Lambda) \times A^c$ and $A^b + (k+1) \cdot \Lambda$ are matrices where each entry is a univariate linear term. The test on line 9 of the Algorithm 5 guarantees that the above equality holds for at least k=0 and k=1. Clearly, if $t_1(0) = t_2(0)$ and $t_1(1) = t_2(1)$ for two univariate linear terms t_1, t_2, then $t_1(k) = t_2(k)$ for all $k \geq 0$. Hence, the MaxPeriodic returns always $L = \infty$. We summarize these observations in the following proposition

Proposition 17. *Given a finite monoid matrix $A \in \mathbb{Z}^{N\times N}$, integers $b \geq 0$, $c > 0$, and a matrix $\Lambda \in \mathbb{Z}^{(N+1)\times(N+1)}$ such that $A_h^{b+c} = A_h^b + \Lambda$ and $A_h^{b+1c} = A_h^{b+c} + \Lambda$, the procedures* MAXCONSISTENT *and* MAXPERIODIC *run in constant time.*

Finally, we prove the asymptotic complexity on the running of Algorithm 5 for a finite monoid affine relation R in terms of its prefix, period, and the number of variables used to define R.

Theorem 12. *Let R be a difference bounds relation with prefix B and period C. Then, Algorithm 5 computes the transitive closure of R in at most $\mathcal{O}((B+C)^2 \cdot N^3)$ time.*

Proof: Let R be a finite monoid affine relation and let R_u be the update of R. If follows from Lemma 16 that asymptotic bound on the time needed to compute the transitive closure of R and R_u are same. Thus, we consider only an update relation in a homogenous form encoded as a matrix $A_h \in \mathbb{Z}_\infty^{(N+1)\times(N+1)}$.

By Theorem 2, Algorithm 5 takes at most $\mathcal{O}((B+C)^2)$ iterations of the main loop and in each iteration and moreover, the algorithm considers a prefix and period candidates b and c such that both b and c are bounded by $\mathcal{O}((B+C)^2)$. By Proposition 17, procedures MAXCONSISTENT and MAXCONSISTENT run in constant time. The test on line 8 amounts to equality of two matrices and can be thus performed in $\mathcal{O}(N^2)$ time. The greatest power of a relation that is computed by the algorithm is R^{b+2c}. Since the composition of an update in a homogenous form with itself amounts to matrix multiplication, if follows that these computations are performed in $\mathcal{O}((B+C) \cdot N^3)$ time. Hence, the total bound on the running time of Algorithm 5 is $\mathcal{O}((B+C)^2 \cdot N^3)$. □

8. Complexity of the Transitive Closure Algorithm

This chapter is concerned with the worst-case complexity of the transitive closure algorithm from Chapter **??** (Algorithm 5) when applied to difference bounds, octagonal, and finite monoid affine relations. For a periodic relation $R \subseteq \mathbb{Z}^N \times \mathbb{Z}^N$ with prefix $b \geq 0$ and period $c > 0$, the asymptotic bound on the running time of Algorithm 5 is $\mathcal{O}((b+c)^8 \cdot \|R\|^3 \cdot N^9)$ if R is a difference bounds or an octagonal relation (by Theorem 8 and 10), where $\|R\|$ denotes the sum of absolute values of the coefficients of R. The asymptotic bound is $\mathcal{O}((b+c)^2 \cdot N^3)$ if R is a finite monoid affine relation (by Theorem 12).

The main issue, dealt with in this chapter, is thus the evaluation of the upper bounds of the prefix b and period c for each of these classes of relations. We prove that for difference bounds relations, b is asymptotically bounded by $\|R\| \cdot 2^{\mathcal{O}(N)}$ and c is bounded by $2^{\mathcal{O}(N)}$. For octagonal relations, the bound on the period is same as for difference bounds relations and the prefix is bounded by $\|R\|^2 \cdot 2^{\mathcal{O}(N)}$. For finite monoid affine relations, $b + c$ is the size of the monoid, which in turn is proved to be bounded by $2^{\mathcal{O}(N^{\log_{10} 11})}$. Columns 2 and 3 in Table 1 summarize these results. Combining the bounds on the size of the prefix and the period with the bounds given by Theorem 8, 10, and 12, we obtain asymptotic bounds on the running time of Algorithm 5 in terms of N and $\|R\|$ (the last column in Table 1).

Table 1: Transitive Closure Complexities for Periodic Relations

CLASS	PREFIX	PERIOD	TRANSITIVE CLOSURE
difference bounds	$\|R\| \cdot 2^{\mathcal{O}(N)}$	$2^{\mathcal{O}(N)}$	$\|R\|^8 \cdot 2^{\mathcal{O}(N)}$
octagonal	$\|R\|^2 \cdot 2^{\mathcal{O}(N)}$	$2^{\mathcal{O}(N)}$	$\|R\|^{16} \cdot 2^{\mathcal{O}(N)}$
finite monoid affine	$2^{\mathcal{O}(N^{\log_{10} 11})}$	$2^{\mathcal{O}(N^{\log_{10} 11})}$	$2^{\mathcal{O}(N^{\log_{10} 11})}$

In all cases, Algorithm 5 runs in EXPTIME in the number of variables, and PTIME in the sum of absolute values of the coefficients of R or, equivalently, in EXPTIME in the size of the binary representation of R.

8.1. Difference Bounds Relations

Any difference bounds relation $R \in \mathcal{R}_{db}$ is periodic, by Theorem 7. This result extends to the octagonal class $\mathcal{R}_{oct}$, by Theorem 9. The periodicity of difference bounds relations is a consequence of the fact that the sequence of tropical matrix powers $\{\mathcal{M}_R^{\boxtimes^i}\}_{i\geq 0}$ where $\mathcal{M}_R$ is the incidence matrix of the common transition table $T_R = (Q, \Delta, w)$ of the zigzag automata defined for R (see Section 6.1.3), is periodic by Theorem 6. Since each power of R is encoded by a matrix which is a projection of a tropical power of $\mathcal{M}_R$, the prefix of R is not greater than the prefix of $\{\mathcal{M}_R^{\boxtimes^i}\}_{i\geq 0}$, while the period of R is a divisor of the period of $\{\mathcal{M}_R^{\boxtimes^i}\}_{i\geq 0}$. In this section, we prove a $\|R\| \cdot 2^{\mathcal{O}(N)}$ upper bound for the prefix and a $2^{\mathcal{O}(N)}$ upper bound for the period of $\mathcal{M}_R$. By the previous arguments, these bounds are also bounds for the prefix and the period of R, respectively.

In the rest of this section, let $\mathbf{x} = \{x_1, \ldots, x_N\}$ be a set of variables, $R(\mathbf{x}, \mathbf{x}')$ be a difference bounds relation, and $T_R = (Q, \Delta, w)$ be the common transition table of zigzag automata defined for R.

8.2. Bounding the Prefix

We start by instantiating Lemma 7 for the common transition table $T_R = (Q, \Delta, w)$ of zigzag automata defined for R.

Corollary 3. *Let $T_R = (Q, \Delta, w)$ be the common transition table of zigzag automata defined for a difference bounds relation R, and $u, v \in Q$ be two control states. Then for every minimal weight path ρ form u to v, such that $|\rho| \geq \|R\| \cdot \|Q\|^6$, there exists a path ρ' from u to v, such that $w(\rho) = w(\rho')$ and $|\rho| = |\rho'|$, and a basic path scheme $\theta = \sigma \cdot \lambda^* \cdot \sigma'$, such that $\rho' = \sigma \cdot \lambda^b \cdot \sigma'$, for some $b \geq 0$. Moreover, there exists $c \mid \frac{lcm(1,\ldots,\|Q\|-1)}{|\lambda|}$ such that $\sigma \cdot \lambda^{b+kc} \cdot \sigma'$ is a minimal weight path from u to v, for all $k \geq 0$.*

Proof: We obtain the statement of the corollary by instantiating Lemma 7 with $T_R = (Q, \Delta, w)$, in which case $\mu(T_R) \leq \|R\|$. □

Similarly, we instantiate Theorem 6.

Corollary 4. *Let $T_R = (Q, \Delta, w)$ be the common transition table of zigzag automata defined for a difference bounds relation R, and let $\mathcal{M}_R$ be its incidence matrix. Then, the sequence $\{\mathcal{M}_R^{\boxtimes^i}\}_{i \geq 0}$ is periodic. Moreover, its prefix b is bounded by $\|R\| \cdot 2^{\mathcal{O}(N)}$, and its period divides $\mathrm{lcm}(1, \ldots, 5^N)$ and is bounded by $2^{2^{\mathcal{O}(N)}}$.*

Proof: We obtain the statement of the corollary by instantiating Theorem 6 with $T_R = (Q, \Delta, w)$, in which case $\mu(T_R) \leq \|R\|$ and $\|Q\| = 5^N$. □

A direct consequence of Corollary 4 is that the prefix of a difference bounds relation R is bounded by $\|R\| \cdot 2^{\mathcal{O}(N)}$.

Corollary 5. *Let $\mathbf{x} = \{x_1, \ldots, x_N\}$ be a set of variables. Given a difference bounds relation $R(\mathbf{x}, \mathbf{x}')$, its prefix is bounded by $\|R\| \cdot 2^{\mathcal{O}(N)}$.*

Proof: We distinguish two cases. First, if R is $*$-consistent, then the bound $\|R\| \cdot 2^{\mathcal{O}(N)}$ on the prefix of $T_R = (Q, \Delta, w)$ that follows from Corollary 4 is also the bound on the prefix of R, by Proposition 10.
If R is not $*$-consistent, there exists a power $\ell > 0$ such that $R^\ell \Leftrightarrow \bot$. Consequently, there exists a minimal weight path ρ of length ℓ in the even zigzag automaton for R, recognizing a negative cycle. By Lemma 6, there exists an equivalent path ρ' of the form $\sigma \cdot \lambda^k \cdot \sigma'$, where $|\sigma \cdot \sigma'| < 5^{4N}$ and $|\lambda| < 5^N$, for some $k \geq 0$. We have $\ell = |\sigma \cdot \sigma'| + k|\lambda|$. The prefix of R is the minimal length ℓ such that $w(\rho) = w(\rho') < 0$. If $w(\sigma \cdot \sigma') < 0$, then this length is $|\sigma\sigma'| < 5^{4N}$. Otherwise, if $w(\sigma \cdot \sigma') \geq 0$, we have $w(\lambda) < 0$, or else ρ' could not encode a negative cycle, independently of how large it is. Then $w(\rho') < 0$ if and only if $k > \frac{w(\sigma \cdot \sigma')}{-w(\lambda)}$ Since $-w(\lambda) > 0$ and $w(\lambda) \in \mathbb{Z}$, we have $-w(\lambda) \geq 1$. A sufficient condition is that $k > \|R\| \cdot 5^{4N} > w(\sigma \cdot \sigma')$, hence

$\ell = |\rho'| > 5^{4N} + \|R\| \cdot 5^{5N}$, i.e. the prefix of a $*$-inconsistent relation R is also asymptotically bounded by $\|R\| \cdot 2^{\mathcal{O}(N)}$. □

Similarly, a direct consequence of Corollary 4 is the bound on the period which is double exponential in N. In the next section, we prove that this bound can be improved to $2^{\mathcal{O}(N)}$.

Corollary 6. *Let $\mathbf{x} = \{x_1, \ldots, x_N\}$ be a set of variables. Given a difference bounds relation $R(\mathbf{x}, \mathbf{x}')$, its period is bounded by $2^{2^{\mathcal{O}(N)}}$.*

Proof: The period of $*$-inconsistent relation R is 1, by Definition 5, which is clearly bounded by $2^{2^{\mathcal{O}(N)}}$.

Let $T_R = (Q, \Delta, w)$ be the common transition table of zigzag automata of a $*$-consistent R. By Corollary 4, the period c of T_R is bounded by $2^{2^{\mathcal{O}(N)}}$. Since each element M_{R^i}, $i \geq 0$, of the sequence $\{M_{R^n}\}_{n \geq 0}$ of powers of R is obtained as a projection of $\mathcal{M}_R^{\boxtimes^i}$, it follows that the period of R divides the period of T_R. Thus, the period of R is bounded by $2^{2^{\mathcal{O}(N)}}$ too. □

8.3. Bounding the Period

In this section, we refine Corollary 6 and show that the period of difference bounds relations is bounded by a single exponential. We start by defining several key notions and giving a high level idea of the proof.

8.3.1. Key Notions and a Proof Idea

Given a difference bounds relation $R(\mathbf{x}, \mathbf{x}')$, $\mathbf{x} = \{x_1, \ldots, x_N\}$ and the unfolded graph $\mathcal{G}_R^\omega$, we define the composition, power, relative length, and relative average weight operators on paths in $\mathcal{G}_R^\omega$.

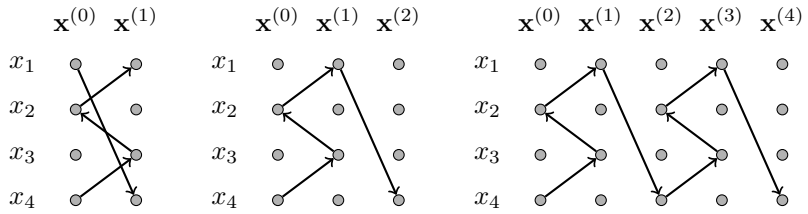
Definition 23. *Let $R(\mathbf{x}, \mathbf{x}')$, $\mathbf{x} = \{x_1, \ldots, x_N\}$, be a difference bounds relation and let*

$$\rho : x_{i_0}^{(l_0)} \to x_{i_1}^{(l_1)} \to \ldots \to x_{i_m}^{(l_m)} \qquad \rho' : x_{j_0}^{(k_0)} \to x_{j_1}^{(k_1)} \to \ldots \to x_{j_n}^{(k_n)}$$

$m, n \geq 1$, be two paths in $\mathcal{G}_R^\omega$. The relative path length operator *is defined as $\|\rho\| = |l_m - l_0|$. If $\|\rho\| > 0$, we define the* relative average weight *operator $\overline{\overline{w}}(\rho) = \frac{w(\rho)}{\|\rho\|}$. If $i_m = j_0$, we define $\rho.\rho'$, the* composition *of ρ with ρ', as*

$$\rho.\rho' = x_{i_0}^{(l_0)} \to x_{i_1}^{(l_1)} \to \ldots \to x_{i_m}^{(l_m)} \to x_{j_1}^{(k_1 - d)} \to \ldots \to x_{j_n}^{(k_n - d)}$$

where $d = k_0 - l_m$. Further, if $i_0 = i_m$, we define ρ^k, $k \geq 1$, the k-th power *of ρ as k-times composition of ρ with itself.*



(a) Edges (b) Repeating path ρ (c) ρ^2

Figure 14: Illustration of repeating path.

Example 8. *Consider the edges in Figure 14(a) and build paths* $\rho_1 : x_4^{(0)} \rightarrow x_3^{(1)} \rightarrow x_2^{(0)} \rightarrow x_1^{(1)}$ *and* $\rho_2 : x_1^{(0)} \rightarrow x_4^{(1)}$. *Their concatenation* $\rho_1.\rho_2$ *results in a path* $\rho : x_4^{(0)} \rightarrow x_3^{(1)} \rightarrow x_2^{(0)} \rightarrow x_1^{(1)} \rightarrow x_4^{(2)}$ *depicted in Figure 14(b). Note that* $\|\rho\| = 2$. *The second power of* ρ, *denoted* ρ^2, *is depicted in Figure 14(c).*

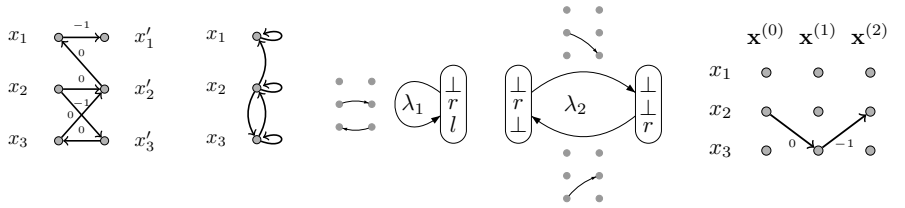
Next, we define several notions characterizing the structure of paths in $\mathcal{G}_R^\omega$.

Definition 24. *Let* $R(\mathbf{x}, \mathbf{x}')$, $\mathbf{x} = \{x_1, \ldots, x_N\}$, *be a difference bounds relation and let* $\rho = x_{i_0}^{(l_0)} \rightsquigarrow x_{i_m}^{(l_m)}$ *be a path in* $\mathcal{G}_R^\omega$. *We say that* ρ *is* forward (fw) *if and only if* $l_m > l_0$. *We say that* ρ *is* backward (bw) *if and only if* $l_m < l_0$. *We say that* ρ *is* repeating *if and only if* $i_0 = i_m$. *We say that* ρ *is* essential *if and only if for all* $1 \leq j < k \leq m$, $i_j = i_k$ *only if* $j = 0$ *and* $k = m$. *Path* ρ *is said to be a cycle if and only* $i_0 = i_m$ *and* $l_0 = l_m$. *We say that* ρ *is cyclic if and only if* ρ *has a subpath that is a cycle. A path is acyclic if and only if it is not cyclic.*

Intuitively, repeating path can composed with itself arbitrary many times. Note that the length of an essential path ρ is at most N, $|\rho| \leq N$. Consequently, its relative length is at most N too, $\|\rho\| \leq N$. Next, we define $\mathcal{G}_R^f$, the *folded graph* of R. Intuitively, $\mathcal{G}_R^f$ projects all edges onto unprimed variables $\mathbf{x}$.

Definition 25. *Let* $R(\mathbf{x}, \mathbf{x}')$, $\mathbf{x} = \{x_1, \ldots, x_N\}$, *be a difference bounds relation and let* $\mathcal{G}_R$ *be its graph representation. The folded graph of* $\mathcal{G}_R$ *is defined as* $\mathcal{G}_R^f = (\mathbf{x}, E)$, *where* $x_i \rightarrow x_j$ *is an edge in* E *if and only if* $x_i \xrightarrow{c} x_j$, $x_i' \xrightarrow{c} x_j'$, $x_i \xrightarrow{c} x_j'$, *or* $x_i' \xrightarrow{c} x_j$ *is an edge of* $\mathcal{G}_R$. *We write* $x_i \sim x_j$ *if and only if* x_i *and* x_j *belong to the same strongly connected component of* $\mathcal{G}_R^f$. *Clearly,* $\sim$ *is an equivalence relation.*

Note that folded graphs are not weighted. Figure 15(b) depicts the folded graph $\mathcal{G}_R^f$ of $\mathcal{G}_R$ from Figure 15(a). The folded graph in Figure 15(b) has two strongly connected components $\{x_1\}$ and $\{x_2, x_3\}$.



(a) $\mathcal{G}_R$ (b) $\mathcal{G}_R^f$ (c) T_R with a simple cycle λ_2 (d) The label of λ_2

Figure 15: Folded graph. Zigzag automaton with a simple cycle.

Then, we observe that each cycle λ in $T_R = (Q, \Delta, w)$ (a *zigzag cycle*, for short) encodes a set of forward $*$-acyclic and backward $*$-acyclic paths.

Definition 26. *A repeating path ρ is* $*$-acyclic *if and only if ρ^k is acyclic for all $k \geq 1$.*

Intuitively, a $*$-acyclic path can be composed with itself arbitrary many times without producing cyclic subpaths. For instance, the path ρ depicted in Figure 14(b) is $*$-acyclic. Note that each essential repeating path is $*$-acyclic.

Further, we study the structure of path schemes $\sigma.\lambda^*.\sigma'$ from Lemma 3 and prove that one can without loss of generality assume that λ is *simple*.

Definition 27. *A cycle λ in the transition table of a zigzag automaton $T_R = (Q, \Delta, w)$ is* simple *if and only if it encodes at most one $*$-acyclic path per equivalence class of $\sim$ relation. A basic path scheme $\theta = \sigma.\lambda^*.\sigma'$ is* simple *if and only if λ is simple.*

Figure 15(c) illustrates a part of the transition table of the zigzag automaton corresponding to the relation in Figure 15(a). The cycle λ_1 depicted in Figure 15(c) is not simple, since it encodes two $*$-acyclic paths $x_2^{(0)} \xrightarrow{0} x_2^{(1)}$ and $x_3^{(0)} \xrightarrow{0} x_3^{(1)}$ from the same strongly connected component $\{x_2, x_3\}$. On the other hand, λ_2 is simple, since it encodes only one $*$-acyclic path $x_2^{(0)} \xrightarrow{0} x_3^{(1)} \xrightarrow{-1} x_2^{(2)}$ depicted in Figure 15(d).

Next, we prove that we can make the statement of Lemma 3 even more accurate and consider, without loss of generality, only path schemes with cycles whose length divides $\mathrm{lcm}(1, \ldots, N)$. Let μ_j be a $*$-acyclic path of the form $\mu_j : x_{i_j} \rightsquigarrow x_{i_j}$ encoded in a simple zigzag cycle λ, where $x_{i_j} \in \mathbf{z}_j$ for some equivalence class $\mathbf{z}_j \in \mathbf{x}_{/\sim}$. We first observe that there exists an essential $*$-acyclic path μ_j of the form $\nu_j : x_{k_j} \rightsquigarrow x_{k_j}$, where $x_{k_j} \in \mathbf{z}_j$ as well. Supposing that λ encodes m $*$-acyclic paths, the intuition is to build a cycle λ' as follows: letting $L = \mathrm{lcm}\{\|\nu_1\|, \ldots, \|\nu_m\|\}$, the cycle λ' will encode paths $\nu_j^{d_j}$, where $d_j = \frac{L}{\|\nu_j\|}$. Then, since $|\lambda'| = L$ and $\|\nu_j\| \leq N$, it follows that the length of λ' divides $\mathrm{lcm}\{1, \ldots, N\}$.

Since μ_j and ν_j belong to the same equivalence class $\mathbf{z}_j$, by the above observations, it follows that there exist *essential* paths $\xi_j : x_{i_j} \rightsquigarrow x_{k_j}$ and $\zeta : x_{k_j} \rightsquigarrow x_{i_j}$. These paths can be used to connect μ_j with ν_j and vice versa. The notion of a *connecting path* captures this idea:

Definition 28. *A forward repeating path $\tau : x_i \rightsquigarrow x_i$ is called a* connecting path *if it is of the form*

$$\tau \quad = \quad \mu^r \quad . \quad \xi \quad . \quad \nu^s \quad . \quad \zeta \quad . \quad \mu^t$$

where

- $\mu : x_i \rightsquigarrow x_i$, $\nu : x_k \rightsquigarrow x_k$ *are forward $*$-acyclic paths, and*
- $\xi : x_i \rightsquigarrow x_k$, $\zeta : x_k \rightsquigarrow x_i$ *are essential paths.*

Note that in a connecting path τ, the repeating paths μ and ν are allowed to be raised to a positive powers r, s, t. Figure 16 illustrates a connecting path.

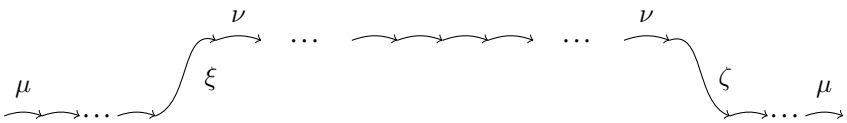


Figure 16: Connecting path τ.

In order to ensure correctness of the construction, we also need to build two zigzag paths π_1 and π_2 that will connect λ with λ' and vice versa, respectively. In other words, we need to ensure that $\lambda.\pi_1.\lambda'.\pi_2.\lambda$ will form a valid zigzag path. To this end, we build connecting paths $\tau_1, \ldots, \tau_m$. Here we encounter a problem of synchronizing the positions at which ν_j appears for the first time in τ_j. This problem is due to the fact that the relative length of ξ_j may be arbitrary – we only know that its relative length is bounded by N, since ξ_j is essential. Lemma 21 proves that this problem can be overcome. Finally, we prove in Lemma 22 the desired claim that we can, without loss of generality, assume path schemes $\sigma.\lambda^*.\sigma'$ where $|\lambda|$ divides $\mathrm{lcm}(1, \ldots, N)$.

Then, we can refine Corollary 6 and establish the upper bound of $2^{O(n)}$ on the period for zigzag automata. Since $M^*_{R^m}$, the encoding of the m-th power of R, is a projection of $\mathcal{M}_R^{\boxtimes^m}$, the bound on the period of the sequence $\{\mathcal{M}_R^{\boxtimes^m}\}_{m\geq 0}$ is a valid bound on the period of the sequence $\{M^*_{R^m}\}_{m\geq 0}$ and consequently, it is a bound on the period of a difference bounds relation R (Theorem 13). In Theorem 14, we show that this result extends rather easily to the period of octagonal relations. Moreover, Theorem 14 shows how the bound of $\|R\|^2 \cdot 2^{\mathcal{O}(N)}$ on the prefix of and octagonal relation can be inferred.

8.3.2. Repeating Paths – Decomposition and Optimality

Repeating essential paths can be seen as building blocks of each repeating path, as the following proposition states. Note that each essential repeating path is either forward, backward or an elementary cycle.

Proposition 18. *Each repeating path ρ can be decomposed into a set of essential repeating paths $\mathcal{F}(\rho)$ such that*

$$w(\rho) \quad = \quad \sum_{\mu\in\mathcal{F}_{\triangleright}(\rho)} w(\mu) + \sum_{\mu\in\mathcal{F}_{\triangleleft}(\rho)} w(\mu) + \sum_{\mu\in\mathcal{F}_{\circ}(\rho)} w(\mu)$$

where $\mathcal{F}_{\triangleright}(\rho), \mathcal{F}_{\triangleleft}(\rho), \mathcal{F}_{\circ}(\rho) \subseteq \mathcal{F}(\rho)$ are the maximal subset of forward, backward, and cyclic paths, respectively. Moreover,

$$\begin{array}{lcll} \|\rho\| & = & \sum_{\mu\in\mathcal{F}_{\triangleright}(\rho)} \|\mu\| - \sum_{\mu\in\mathcal{F}_{\triangleleft}(\rho)} \|\mu\| & \textit{if } \rho \textit{ is forward,} \\ \|\rho\| & = & \sum_{\mu\in\mathcal{F}_{\triangleleft}(\rho)} \|\mu\| - \sum_{\mu\in\mathcal{F}_{\triangleright}(\rho)} \|\mu\| & \textit{if } \rho \textit{ is backward,} \\ \|\rho\| & = & \sum_{\mu\in\mathcal{F}_{\triangleleft}(\rho)} \|\mu\| = \sum_{\mu\in\mathcal{F}_{\triangleright}(\rho)} \|\mu\| = 0 & \textit{if } \rho \textit{ is a cycle.} \end{array}$$

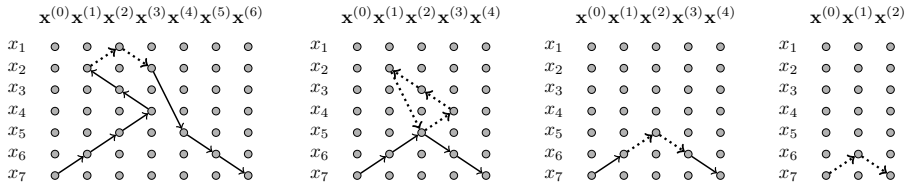
Proof: Let ρ be arbitrary path and denote $\rho_0 = \rho$. For each $i \geq 0$, we define ρ_{i+1} inductively as follows. Let μ_i be an arbitrary essential repeating subpath of ρ_i, i.e. $\rho_i = \theta_i.\mu_i.\theta'_i$ for some θ_i, θ'_i. Then, construct ρ_{i+1} by erasing μ_i from ρ_i, i.e. $\rho_{i+1} = \theta_i.\theta'_i$. Clearly, this decomposition terminates since ρ_{k+1} is empty for some $k \geq 0$. Then, $\mathcal{F}(\rho) = \{\mu_0, \ldots, \mu_k\}$. Next, let us define $D_i \in \mathbb{Z}$, $i = k+1, \ldots, 0$ inductively as follows: $D_{k+1} = 0$ and for each $i = k, \ldots, 0$, define

$$\begin{array}{ll} D_i = D_{i+1} + \|\mu_i\| & \text{if } \mu_i \text{ is forward,} \\ D_i = D_{i+1} - \|\mu_i\| & \text{if } \mu_i \text{ is backward,} \\ D_i = D_{i+1} & \text{if } \mu_i \text{ is an elementary cycle.} \end{array}$$

Clearly, for each $1 \leq i \leq k$,

$$\begin{array}{lllll} \|\rho_i\| = D_i & \textbf{iff} & \rho_i \text{ is forward} & \textbf{iff} & D_i > 0, \\ \|\rho_i\| = -D_i & \textbf{iff} & \rho_i \text{ is backward} & \textbf{iff} & D_i < 0, \\ \|\rho_i\| = 0 & \textbf{iff} & \rho_i \text{ is a cycle} & \textbf{iff} & D_i = 0. \end{array}$$

Recall that $\rho = \rho_0$. Thus, if ρ if forward, then $\|\rho\| = \sum_{\mu\in\mathcal{F}_{\triangleright}(\rho)} \|\mu\| - \sum_{\mu\in\mathcal{F}_{\triangleleft}(\rho)} \|\mu\|$ and if ρ is backward, then $\|\rho\| = \sum_{\mu\in\mathcal{F}_{\triangleleft}(\rho)} \|\mu\| - \sum_{\mu\in\mathcal{F}_{\triangleright}(\rho)} \|\mu\|$. Clearly, if ρ is a cycle, then $\|\rho\| = \sum_{\mu\in\mathcal{F}_{\triangleleft}(\rho)} \|\mu\| = \sum_{\mu\in\mathcal{F}_{\triangleright}(\rho)} \|\mu\| = 0$. □



(a) ρ_0 and its subpath μ_0 (b) ρ_1, μ_1 (c) ρ_2, μ_2 (d) ρ_3, μ_3

Figure 17: Decomposition of a repeating path to essential repeating paths.

Example 9. *Consider a path ρ depicted in Figure 17(a). Figures 17(a-d) illustrate a decomposition of ρ into a set of essential repeating paths. Essential subpaths $\mathcal{F}(\rho) = \{\mu_0, \ldots, \mu_3\}$ selected during the decomposition are dotted.*

We next show that the average weight of a repeating path ρ is equal to the average weight of an arbitrary power of ρ.

Proposition 19. *Let ρ be a repeating path and let $d \geq 1$. Then, $\overline{\overline{w}}(\rho^d) = \overline{\overline{w}}(\rho)$.*

Proof: Observe that $\overline{\overline{w}}(\rho^d) = \frac{d \cdot w(\rho)}{d \cdot \|\rho\|} = \frac{w(\rho)}{\|\rho\|} = \overline{\overline{w}}(\rho)$. □

Next, we introduce a notion of *optimal* path.

Definition 29. *Let $\mathbf{z} \in \mathbf{x}_{/\sim}$ be an equivalence class of $\sim$ and let $S^{\mathbf{z}}_{\triangleright}$ ($S^{\mathbf{z}}_{\triangleleft}$) be the set of all forward (backward) repeating paths in $\mathcal{G}_R$ of the form $x_i \rightsquigarrow x_i$, for some $x_i \in \mathbf{z}$. A path $\rho \in S^{\mathbf{z}}_{\triangleright}$ is* $\triangleright$-optimal *if and only if $\overline{\overline{w}}(\rho) \leq \overline{\overline{w}}(\rho)$ for all $\rho' \in S^{\mathbf{z}}_{\triangleright}$. Similarly, a path $\rho \in S^{\mathbf{z}}_{\triangleleft}$ is* $\triangleleft$-optimal *if and only if $\overline{\overline{w}}(\rho) \leq \overline{\overline{w}}(\rho)$ for all $\rho' \in S^{\mathbf{z}}_{\triangleleft}$.*

We next show that the average weight of optimal paths are determined by average weights of *critical* essential repeating paths. Thus, we first characterize these paths. For each equivalence class $\mathbf{z} \in \mathbf{x}_{/\sim}$, we define the set of essential repeating forward paths $P_{\triangleright}(\mathbf{z})$, minimal average weight of these paths $C_{\triangleright}(\mathbf{z})$, and a subset of *fw-critical* paths $P^c_{\triangleright}(\mathbf{z})$ as follows. Note that we allow a path to cross nodes $\mathbf{x}^{(\ell)}$, where $\ell < 0$, for notational convenience.

$$\begin{array}{rcl} P_{\triangleright}(\mathbf{z}) & = & \{\rho : x_i^{(\ell)} \rightsquigarrow x_i^{(\ell')} \mid \ell' > \ell, \rho \text{ is an essential, repeating path in } \mathcal{G}_R^m, m \geq 0\} \\ C_{\triangleright}(\mathbf{z}) & = & \min\{\|\rho\| \mid \rho \in P_{\triangleright}(\mathbf{z})\} \\ P^c_{\triangleright}(\mathbf{z}) & = & \{\rho \in P_{\triangleright}(\mathbf{z}) \mid \|\rho\| = C_{\triangleright}(\mathbf{z})\} \end{array}$$

Similarly, we define $P_{\triangleleft}(\mathbf{z}), C_{\triangleleft}(\mathbf{z}), P^c_{\triangleleft}(\mathbf{z})$ for backward paths.

The following lemma gives a precise characterization of optimal paths, based on properties of critical paths defined above.

Lemma 17. *Let $\mathbf{z} \subseteq \mathbf{x}$ be an equivalence class of $\sim$ and let $\rho : x \rightsquigarrow x$, $x \in \mathbf{z}$, be a repeating path in $\mathcal{G}_R$.*

1. *If ρ is forward, then $\overline{\overline{w}}(\rho) \geq C_{\triangleright}(\mathbf{z})$. Moreover, ρ is $\triangleright$-optimal if and only if $\overline{\overline{w}}(\rho) = C_{\triangleright}(\mathbf{z})$ if and only if*
   (a) *$\overline{\overline{w}}(\mu) = C_{\triangleright}(\mathbf{z})$ for each forward path $\mu \in \mathcal{F}(\rho)$,*
   (b) *$\overline{\overline{w}}(\mu) = -C_{\triangleright}(\mathbf{z})$ for each backward path $\mu \in \mathcal{F}(\rho)$,*
   (c) *$w(\mu) = 0$ for each cycle $\mu \in \mathcal{F}(\rho)$.*

   Moreover, if $\mathcal{F}(\rho)$ contains a backward path and $\overline{\overline{w}}(\rho) = C_{\triangleright}(\mathbf{z})$, then $C_{\triangleright}(\mathbf{z}) = -C_{\triangleleft}(\mathbf{z})$.
2. *If ρ is backward, then $\overline{\overline{w}}(\rho) \geq C_{\triangleleft}(\mathbf{z})$. Moreover, ρ is $\triangleleft$-optimal if and only if $\overline{\overline{w}}(\rho) = C_{\triangleleft}(\mathbf{z})$ if and only if*
   (a) *$\overline{\overline{w}}(\mu) = -C_{\triangleleft}(\mathbf{z})$ for each forward path $\mu \in \mathcal{F}(\rho)$,*
   (b) *$\overline{\overline{w}}(\mu) = C_{\triangleleft}(\mathbf{z})$ for each backward path $\mu \in \mathcal{F}(\rho)$,*
   (c) *$w(\mu) = 0$ for each cycle $\mu \in \mathcal{F}(\rho)$.*

   Moreover, if $\mathcal{F}(\rho)$ contains a forward path and $\overline{\overline{w}}(\rho) = C_{\triangleleft}(\mathbf{z})$, then $C_{\triangleleft}(\mathbf{z}) = -C_{\triangleright}(\mathbf{z})$.
3. *If ρ is a cycle, then $\overline{\overline{w}}(\rho) \geq 0$. Moreover, $w(\rho) = 0$ if and only if*
   (a) *$\overline{\overline{w}}(\mu) = C_{\triangleright}(\mathbf{z}) = -C_{\triangleleft}(\mathbf{z})$ for each forward path $\mu \in \mathcal{F}(\rho)$,*
   (b) *$\overline{\overline{w}}(\mu) = C_{\triangleleft}(\mathbf{z}) = -C_{\triangleright}(\mathbf{z})$ for each backward path $\mu \in \mathcal{F}(\rho)$,*
   (c) *$w(\mu) = 0$ for each cycle $\mu \in \mathcal{F}(\rho)$.*

Proof: We give the proof for the case when ρ is forward. Proofs for other cases are similar.

Let $S_\triangleright, S_\triangleleft, S_\circ \subseteq \mathcal{F}(\rho)$ be the sets of all forward paths, backward paths, and cycles in $\mathcal{F}(\rho)$, respectively. Clearly, $\overline{\overline{w}}(\mu) \geq C_\triangleright(\mathbf{z})$ for each $\mu \in S_\triangleright$. As a corollary of Lemma 18, $C_\triangleleft(\mathbf{z}) + C_\triangleright(\mathbf{z}) \geq 0$ and thus, $\overline{\overline{w}}(\mu) \geq C_\triangleleft(\mathbf{z}) \geq -C_\triangleright(\mathbf{z})$ for each $\mu \in S_\triangleleft$. Since R is $*$-consistent, then $w(\nu) \geq 0$ for each $\mu \in S_\circ$. Thus, for each $\mu \in S_\triangleright$, there exists $d_\mu \geq 0$ such that $\overline{\overline{w}}(\mu) = C_\triangleright(\mathbf{z}) + d_\mu$. Similarly, for each $\mu \in S_\triangleleft$, there exists $d_\mu \geq 0$ such that $\overline{\overline{w}}(\mu) = -C_\triangleright(\mathbf{z}) + d_\mu$. Let us define:

$$w(S_\triangleright) = \sum_{\mu \in S_\triangleright} w(\nu) \qquad w(S_\triangleleft) = \sum_{\mu \in S_\triangleleft} w(\nu) \qquad w(S_\circ) = \sum_{\mu \in S_\circ} w(\mu)$$

We derive:

$$\begin{array}{rcl} w(S_\triangleright) & = & \sum_{\mu \in S_\triangleright} w(\nu) = \sum_{\mu \in S_\triangleright} \overline{\overline{w}}(\mu)\|\mu\| = \sum_{\mu \in S_\triangleright} (C_\triangleright(\mathbf{z}) + d_\mu)\|\mu\| \\ & = & C_\triangleright(\mathbf{z}) \sum_{\mu \in S_\triangleright} \|\mu\| + \sum_{\mu \in S_\triangleright} d_\mu \|\mu\| \\ w(S_\triangleleft) & = & \sum_{\mu \in S_\triangleleft} w(\nu) = \sum_{\mu \in S_\triangleleft} \overline{\overline{w}}(\mu)\|\mu\| = \sum_{\mu \in S_\triangleleft} (-C_\triangleright(\mathbf{z}) + d_\mu)\|\mu\| \\ & = & -C_\triangleright(\mathbf{z}) \sum_{\mu \in S_\triangleright} \|\mu\| + \sum_{\mu \in S_\triangleright} d_\mu \|\mu\| \end{array}$$

Observe that:

$$\begin{array}{rcl} w(S_\triangleright) + w(S_\triangleleft) & = & C_\triangleright(\mathbf{z}) \left(\sum_{\mu \in S_\triangleright} \|\mu\| - \sum_{\mu \in S_\triangleleft} \|\mu\| \right) + \sum_{\mu \in S_\triangleright \cup S_\triangleleft} d_\mu \|\mu\| \\ & = & C_\triangleright(\mathbf{z})\|\rho\| + \sum_{\mu \in S_\triangleright \cup S_\triangleleft} d_\mu \|\mu\| \end{array}$$

The last equality holds since $\|\rho\| = \sum_{\mu \in S_\triangleright} \|\mu\| - \sum_{\mu \in S_\triangleleft} \|\mu\|$. Since $w(\rho) = w(S_\triangleright) + w(S_\triangleleft) + w(S_\circ)$, we infer that

$$\begin{array}{rcl} w(\rho) & = & w(S_\triangleright) + w(S_\triangleleft) + w(S_\circ) \\ & = & C_\triangleright(\mathbf{z})\|\rho\| + \sum_{\mu \in S_\triangleright} d_\mu \|\mu\| + \sum_{\mu \in S_\triangleleft} d_\mu \|\mu\| + \sum_{\mu \in S_\circ} w(\mu) \end{array}$$

Consequently,

$$\overline{\overline{w}}(\rho) \quad = \quad C_\triangleright(\mathbf{z}) + \frac{\sum_{\mu \in S_\triangleright} d_\mu \|\mu\| + \sum_{\mu \in S_\triangleleft} d_\mu \|\mu\| + \sum_{\mu \in S_\circ} w(\mu)}{\|\rho\|}.$$

Since the fraction in the above equation is non-negative, then $\overline{\overline{w}}(\rho) \geq C_\triangleright(\mathbf{z})$. Moreover, $\overline{\overline{w}}(\rho) = C_\triangleright(\mathbf{z})$ if and only if $d_\nu = 0$ for each $\nu \in S_\triangleright$, $d_\nu = 0$ for each $\nu \in S_\triangleleft$, and $w(\nu) = 0$ for each $\nu \in S_\circ$ if and only if $\overline{\overline{w}}(\nu) = C_\triangleright(\mathbf{z})$ for each $\nu \in S_\triangleright$, $\overline{\overline{w}}(\nu) = -C_\triangleright(\mathbf{z})$ for each $\nu \in S_\triangleleft$, and $w(\nu) = 0$ for each $\nu \in S_\circ$.

Suppose that $\mathcal{F}_\triangleleft(\rho) \neq \emptyset$ and $\overline{\overline{w}}(\rho) = C_\triangleright(\mathbf{z})$. Recall that $\overline{\overline{w}}(\mu) \geq C_\triangleleft(\mathbf{z}) \geq -C_\triangleright(\mathbf{z})$ for each $\mu \in S_\triangleleft$. By the above arguments, $\overline{\overline{w}}(\mu) = -C_\triangleright(\mathbf{z})$. Thus, $-C_\triangleright(\mathbf{z}) \geq C_\triangleleft(\mathbf{z}) \geq -C_\triangleright(\mathbf{z})$ and consequently, $C_\triangleleft(\mathbf{z}) = -C_\triangleright(\mathbf{z})$. □

The following technical proposition is later used for proving properties of connecting paths.

Proposition 20. *Let $\mathbf{z} \subseteq \mathbf{x}$ be an equivalence class of $\sim$ and let $\rho : x_i \rightsquigarrow x_i$, $x_i \in \mathbf{z}$, be an optimal forward repeating path in $\mathcal{G}_R$. Then, there exist an optimal essential forward path $\rho' : x_j \rightsquigarrow x_j$ and essential paths $\xi : x_i \rightsquigarrow x_j$, $\zeta : x_j \rightsquigarrow x_i$ such that*

- $\overline{\overline{w}}(\xi.\zeta) = C_{\triangleright}(\mathbf{z})$ *if* $\xi.\zeta$ *is forward,*
- $\overline{\overline{w}}(\xi.\zeta) = -C_{\triangleright}(\mathbf{z})$ *if* $\xi.\zeta$ *is backward,*
- $w(\xi.\zeta) = 0$ *if* $\xi.\zeta$ *forms a cycle.*

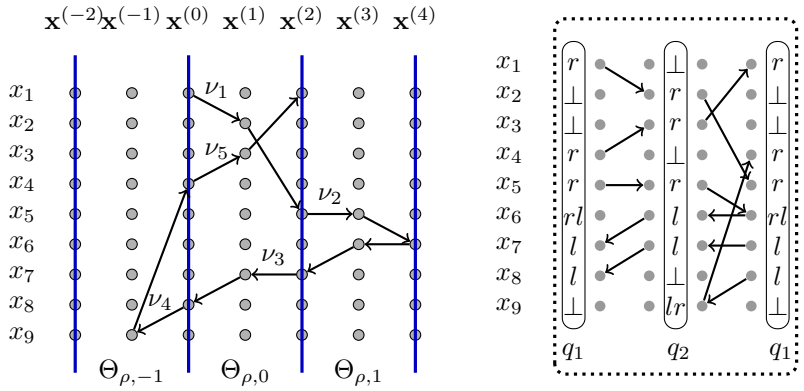
Moreover, $\rho' \in \mathcal{F}(\rho)$ *and* $\mathcal{F}(\xi.\zeta) \subseteq \mathcal{F}(\rho)$.

Proof: Let $\rho_0, \ldots, \rho_k$ and $\mu_0, \ldots, \mu_k \subseteq \mathcal{F}(\rho)$ be paths constructed in a decomposition of ρ into essential repeating paths as in the proof of Proposition 18. Clearly, there exists $0 \leq m \leq k$ such that $\mu_m : x_j \rightsquigarrow x_j$ is forward. Let θ, θ' be paths such that $\rho_m = \theta.\mu_m.\theta'$. Let us decompose θ by erasing its essential repeating subpaths, obtaining $\theta_0, \ldots, \theta_\ell$. Similarly, we decompose θ' and obtain $\theta'_0, \ldots, \theta'_{\ell'}$. Note that we can without loss of generality assume that $\rho_{m+n} = \theta_n.\theta'$ for all $0 \leq n \leq \ell$ and that $\rho_{m+\ell+n} = \theta_\ell.\theta'_n$ for all $0 \leq n \leq \ell'$. Let $\xi = \theta_\ell, \zeta = \theta'_{\ell'}$. Since ρ is fw-optimal, then clearly $\mathcal{F}(\xi.\zeta) = \mathcal{F}(\theta_\ell.\theta'_{\ell'}) \subseteq \mathcal{F}(\rho)$. Applying Lemma 17, we get the remaining properties of $\theta_\ell.\theta'_{\ell'}$ stated in this proposition. □

8.3.3. Anatomy of Zigzag Cycles

We now inspect the structure of cycles in zigzag automata. In particular, we show that each $*$-acyclic path encoded in a zigzag cycle is a concatenation of several *zigzag-segments*:

Definition 30. *Let* $\lambda = q_0 \xrightarrow{G_1} q_1 \xrightarrow{G_2} \ldots \xrightarrow{G_p} q_p$, *where* $q_0 = q_p$, *be a zigzag cycle of length* $|\lambda| = p$, *where* $G_1, \ldots, G_p$ *are subgraphs of* $\mathcal{G}_R$ *that label edges appearing in* λ. *Let* G *be a subgraph of* $\mathcal{G}_R^p$ *constructed as* $G = G_1.G_2 \ldots G_p$. *Each path* θ *in* G *that is maximal in its length is called a* zigzag-segment.



(a) path ρ (b) zigzag cycle λ_ρ

Figure 18: Segmentation of a repeating path to zigzag-segments (a) and construction of a corresponding zigzag cycle (b).

Given a zigzag cycle λ of length $|\lambda| = p$, we write its segments as paths of the form $x_i^{(0)} \rightsquigarrow x_j^{(p)}$, $x_i^{(p)} \rightsquigarrow x_j^{(0)}$, $x_i^{(0)} \rightsquigarrow x_j^{(0)}$, or $x_i^{(p)} \rightsquigarrow x_j^{(p)}$.

Example 10. *Consider a zigzag cycle λ_ρ depicted in Figure 18(b). λ_ρ has five zigzag-segments:*

$$\nu_1 : x_1^{(0)} \to x_2^{(1)} \to x_5^{(2)} \qquad \nu_2 : x_5^{(0)} \to x_5^{(1)} \to x_6^{(2)} \to x_6^{(1)} \to x_7^{(0)}$$
$$\nu_3 : x_7^{(2)} \to x_7^{(1)} \to x_8^{(0)} \qquad \nu_4 : x_8^{(2)} \to x_9^{(1)} \to x_4^{(2)} \qquad \nu_5 : x_4^{(0)} \to x_3^{(1)} \to x_1^{(2)}$$

□

Each zigzag cycle λ, $|\lambda| = p$, encodes a set of forward $*$-acyclic paths of the form $x_i^{(0)} \rightsquigarrow x_i^{(p)}$ and a set of backward $*$-acyclic paths of the form $x_i^{(p)} \rightsquigarrow x_i^{(0)}$. For simplicity, let us first examine cycles which encode one forward $*$-acyclic path $\rho : x_{i_0}^{(\ell_0)} \to \ldots \to x_{i_m}^{(\ell_m)}$, $i_0 = i_m$, $p = \ell_m - \ell_0$, and no backward path. We will describe how λ_ρ, a unique zigzag cycle that encodes ρ, can be built. We first define:

$$L_\rho = |\min\{\ell_k - \ell_0 \mid 0 \le k \le m\}| \qquad \overline{L}_\rho = \left\lceil \frac{L_\rho}{\|\rho\|} \right\rceil$$
$$R_\rho = |\max\{\ell_k - \ell_0 \mid 0 \le k \le m\}| \qquad \overline{R}_\rho = \left\lceil \frac{R_\rho}{\|\rho\|} \right\rceil$$

Intuitively, L_ρ (R_ρ) is the left (right) extent of ρ relative to $x_{i_0}^{(\ell_0)}$.

Example 11. *(ctd.) Given a path $\rho : x_1^{(0)} \rightsquigarrow x_1^{(2)}$ depicted in Figure 18(a), we compute:*

$$L_\rho = |\min\{-1, \ldots, 3\}| = 1 \quad \overline{L}_\rho = \lceil \tfrac{1}{2} \rceil = 1$$
$$R_\rho = |\max\{-1, \ldots, 3\}| = 3 \quad \overline{R}_\rho = \lceil \tfrac{3}{2} \rceil = 2$$

□

Next, let decompose ρ into zigzag segments in the following manner. For each $-\overline{L}_\rho \le j < \overline{R}_\rho$, we define $\Theta_{\rho,j}$, the set of maximal (in their length) subpath of ρ which cross only variables

$$\{\mathbf{x}^{(k)} \mid j \cdot \|\rho\| \le k \le (j+1) \cdot \|\rho\|\}.$$

Then, λ_ρ consists of zigzag-segments $\bigcup_{k=-\overline{L}_\rho}^{\overline{R}_\rho - 1} \Theta_{\rho,k}$. Similar definitions can be made for backward paths. This construction can be generalized for zigzag cycles encoding a set of forward and backward paths.

Example 12. *(ctd.) Given a path $\rho : x_1^{(0)} \rightsquigarrow x_1^{(2)}$ depicted in Figure 18(a), we computed $L_\rho = 1$, $\overline{L}_\rho = 1$, $R_\rho = 3$, $\overline{R}_\rho = 2$. Then, $\Theta_{\rho,-1}$ is a set of maximal subpaths of ρ crossing only $\mathbf{x}^{(-2)} \cup \mathbf{x}^{(-1)} \cup \mathbf{x}^{(0)}$, thus $\Theta_{\rho,-1} = \{\nu_4\}$. Similarly, $\Theta_{\rho,0}$ is a set of maximal subpaths of ρ crossing only $\mathbf{x}^{(0)} \cup \mathbf{x}^{(1)} \cup \mathbf{x}^{(2)}$, thus $\Theta_{\rho,0} = \{\nu_1, \nu_3, \nu_5\}$. Finally, $\Theta_{\rho,1}$ is a set of maximal subpaths of ρ crossing only $\mathbf{x}^{(2)} \cup \mathbf{x}^{(3)} \cup \mathbf{x}^{(4)}$, thus $\Theta_{\rho,1} = \{\nu_2\}$. Clearly, the zigzag cycle in Figure 18(b) encodes segments $\Theta_{\rho,-1} \cup \Theta_{\rho,0} \cup \Theta_{\rho,1}$.* □

The following proposition states that the average weight of a cycle λ in zigzag automaton is the sum of average weight of $*$-acyclic paths that are encoded in the label of λ.

Proposition 21. *Let λ be a zigzag cycle that encodes $*$-acyclic paths $\rho_1, \ldots, \rho_n$. Then $\overline{w}(\lambda) = \overline{\overline{w}}(\rho_1) + \cdots + \overline{\overline{w}}(\rho_n)$.*

Proof: Since $|\lambda| = \|\rho_1\| = \cdots = \|\rho_n\|$, we infer:

$$\overline{w}(\lambda) \quad = \quad \frac{w(\rho_1)+\cdots+w(\rho_n)}{|\lambda|} = \sum_{i=1}^{n} \frac{w(\rho_i)}{|\lambda|} = \sum_{i=1}^{n} \frac{w(\rho_i)}{\|\rho_i\|} = \sum_{i=1}^{n} \overline{\overline{w}}(\rho_i)$$

□

8.3.4. Basic Path Schemes with Simple Zigzag Cycles

This section refines the statement of Lemma 3 by proving that the cycle λ from each basic path scheme can be assumed to be simple, without loss of generality. The next lemma proves that the sum of average weights of a forward repeating and a backward repeating path in $\mathcal{G}_R^m$, $m \geq 1$, from the same equivalence class of the $\sim$ relation is non-negative, whenever R is $*$-consistent.

Lemma 18. *Let R be a $*$-consistent difference bounds relation and let $\rho_i = x_i \rightsquigarrow x_i$ be a forward repeating and $\rho_j = x_j \rightsquigarrow x_j$ be a backward repeating path in $\mathcal{G}_R$ such that $x_i \sim x_j$. Then, $\overline{\overline{w}}(\rho_i) + \overline{\overline{w}}(\rho_j) \geq 0$.*

Proof: Suppose that $\overline{\overline{w}}(\rho_i) + \overline{\overline{w}}(\rho_j) < 0$. Let us define:

$$p = \mathrm{lcm}(\|\rho_i\|, \|\rho_j\|), \;\; d_i = \frac{p}{\|\rho_i\|}, \;\; d_j = \frac{p}{\|\rho_j\|}, \;\; \gamma_i = (\rho_i)^{d_i}, \;\; \gamma_j = (\rho_j)^{d_j}.$$

By Proposition 19, $\overline{\overline{w}}(\rho_i) = \overline{\overline{w}}(\gamma_i)$ and $\overline{\overline{w}}(\rho_j) = \overline{\overline{w}}(\gamma_j)$. Thus, $\overline{\overline{w}}(\gamma_i) + \overline{\overline{w}}(\gamma_j) < 0$. Furthermore, since $\|\gamma_i\| = \|\gamma_j\| = p$, then $p \cdot \overline{w}(\gamma_i) + p \cdot \overline{w}(\gamma_j) = w(\gamma_i) + w(\gamma_j) < 0$. Since $x_i \sim x_j$, there exist essential paths

$$\theta_{ij} = x_i^{(0)} \rightsquigarrow x_j^{(q)} \text{ and } \theta_{ji} = x_j^{(0)} \rightsquigarrow x_i^{(r)}$$

where $0 \leq \|Q\|, |r| < N$. Let $n \geq 0$ be a parameter. We build (refer to Figure 19)

$$\xi = \gamma_i^n.\theta_{ij}.\gamma_j^{2n}.\theta_{ji}$$

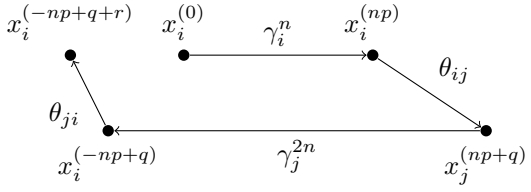


Figure 19: Building ξ

Clearly, ξ is of the form $\xi : x_i^{(0)} \rightsquigarrow x_i^{(-np+q+r)}$. By choosing $n > \lceil \frac{q+r}{p} \rceil$, we make sure that $-np + r + s < 0$. We repeat the path p-times and obtain $\xi^p : x_i^{(0)} \rightsquigarrow x_i^{(p(-np+q+r))}$. Since $|\gamma_i| = p$ and p divides $p(-np+q+r)$, we build $\zeta = \gamma_i^{(np-r-s)}$ which is of the form $\zeta : x_i^{(p(-np+q+r))} \rightsquigarrow x_i^{(0)}$. Clearly, $\xi^p.\zeta$ forms a cycle with weight

$$np \cdot w(\gamma_i) + p \cdot w(\theta_{ij}) + 2np \cdot w(\gamma_j) + p \cdot w(\theta_{ji}) + (np - q - r) \cdot w(\gamma_i)$$

which simplifies to

$$2np \cdot (w(\gamma_i) + w(\gamma_j)) - (q + r) \cdot w(\gamma_i) + p \cdot (w(\theta_{ij}) + w(\theta_{ji})).$$

Since we assumed that $w(\gamma_i) + w(\gamma_j) < 0$, by choosing a sufficiently large n, we obtain a negative cycle in $\mathcal{G}_R^\omega$. Thus, R is not $*$-consistent, contradiction. □

We continue with a technical lemma.

Lemma 19. *Let $G = \langle V, E, w\rangle$ be a weighted digraph, and $u, v \in V$ be two vertices. Let $\theta_1 = \sigma_1.\lambda_1^*.\sigma_1'$ be a basic path scheme and $\rho_1 = \sigma_1.\lambda_1^{b_1}.\sigma_1'$, $b_1 \geq 0$ be a minimal path from u to v such that $|\rho_1| \geq \|V\|^4$. Further, let $\theta_2 = \sigma_2\lambda_2^*.\sigma_2'$ be a path scheme (not necessarily basic), and $\rho_1' = \sigma_1.\lambda_1^{b_1'}.\sigma_1'$ and $\rho_2' = \sigma_2\lambda_2^{b_2'}.\sigma_2'$, where $b_1' > b_1$ and $b_2' \geq 0$, be paths from u to v, and let $L > 0$ be integer such that*

$$\begin{array}{ll} |\rho_1| + L = |\rho_1'| = |\rho_2'|, & \overline{w}(\lambda_1) = \overline{w}(\lambda_2), \\ |\lambda_1| \text{ and } |\lambda_2| \text{ divide } L, & w(\rho_1') = w(\rho_2'). \end{array}$$

Then, there exists a basic path scheme $\theta_3 = \sigma_3.\lambda_2^*.\sigma_3'$ and a path $\rho_3 = \sigma_3.\lambda_2^{b_3}.\sigma_3'$, $b_3 \geq 0$ from u to v such that $w(\rho_3) = w(\rho_1)$ and $|\rho_3| = |\rho_1|$.

Proof: We can use the same techniques as in the proofs of Lemma 6 and 7 and for a given path ρ_2', we construct a basic path scheme $\theta_3 = \sigma_3.\lambda_2^*.\sigma_3'$ and a path $\rho_3' = \sigma_3.\lambda_2^{b_3'}.\sigma_3'$, $b_3' \geq 0$ from u to v such that $|\rho_3'| = |\rho_2'|$ and $w(\rho_3') \leq w(\rho_2')$. Thus, $w(\rho_2') = w(\rho_3') + D$ for some $D \geq 0$. Observe that

$$|\sigma_3.\sigma_3'| \leq \|V\|^4 \leq |\rho_1|.$$

By requirements of the lemma and by construction of ρ_3', $|\rho_1| + L = |\rho_1'| = |\rho_2'| = |\rho_3'|$. Since $|\lambda_2|$ divides L, there exists a path $\rho_3 = \sigma_3.\lambda_2^{b_3}.\sigma_3'$, where $b_3 \geq 0$, such that $|\rho_3| = |\rho_1|$. Furthermore,

$$\begin{array}{rcl} w(\rho_1) & = & w(\rho_1') - \overline{w}(\lambda_1) \cdot L, \text{ and} \\ w(\rho_3) & = & w(\rho_3') - \overline{w}(\lambda_2) \cdot L. \end{array}$$

The lemma requires that $w(\rho_1') = w(\rho_2')$. Combining it with $w(\rho_2') = w(\rho_3') + D$, we obtain that $w(\rho_1') = w(\rho_3') + D$. The lemma also requires that $\overline{w}(\lambda_2) = \overline{w}(\lambda_1)$. Thus,

$$w(\rho_1') - \overline{w}(\lambda_1) \cdot L = w(\rho_3') - \overline{w}(\lambda_2) \cdot L + D$$

and consequently, $w(\rho_1) = w(\rho_3) + D$. Clearly, $D > 0$ would contradict that ρ_1 is minimal, thus we conclude that $D = 0$ and thus $w(\rho_1) = w(\rho_3)$. □

We finally prove the existence of a basic path scheme where the cycle λ is simple.

Lemma 20. *Let $T_R = (Q, \Delta, w)$ be the common transition table of zigzag automata defined for a difference bounds relation $R(\mathbf{x}, \mathbf{x}')$, and $u, v \in Q$ be two control states. Then for every minimal weight path ρ form u to v, such that $|\rho| \geq \|R\| \cdot \|Q\|^6$, there exists a path ρ' from u to v, such that $w(\rho) = w(\rho')$ and $|\rho| = |\rho'|$, and a basic path scheme $\theta = \sigma \cdot \lambda^* \cdot \sigma'$, such that λ is simple, $\rho' = \sigma \cdot \lambda^b \cdot \sigma'$, for some $b \geq 0$. Moreover, there exists $c \mid \frac{lcm(1,\ldots,\|Q\|-1)}{|\lambda|}$ such that $\sigma \cdot \lambda^{b+kc} \cdot \sigma'$ is a minimal weight path from u to v, for all $k \geq 0$.*

Proof: By Lemma 3, there exists a basic path scheme $\theta_1 = \sigma_1\lambda_1^*\sigma_1'$ and a path $\rho_1 = \sigma_1\lambda_1^{b_1}\sigma_1'$, $b_1 \geq 0$ from u to v that have all the properties of this lemma except that λ_1 might not be simple. First, we build a path scheme $\theta_2 = \sigma_2\lambda_2^*\sigma_2'$ which might not be basic, but where λ_2 is simple, $w(\lambda_2) = w(\lambda_1)$, $|\lambda_2| = |\lambda_1|$, and $w(\sigma_1.\lambda_1^{b_1+q+k}) = w(\sigma_2.\lambda_2^k.\sigma_2')$ for some $q > 0$ and for all $k \geq 0$. In other words, if θ_1 is followed by a minimal path of length $L \geq |\sigma_2.\sigma_2'|$, then θ_2 is followed by a minimal path of length L too.

Let $\mathbf{z}_1, \ldots, \mathbf{z}_m$ be equivalence classes of $\sim$. Let q be a control state of the zigzag automaton such that λ_1 is a cycle that starts and ends in q. Let $L, R \subset \{1, \ldots, N\}$ be the set of l-indices and r-indices of q, respectively, and partition them according to equivalence classes $\mathbf{z}_1, \ldots, \mathbf{z}_m$, thus obtaining $L_1, \ldots, L_m$ and $R_1, \ldots, R_m$. For each $1 \leq i \leq m$, let $\rho_{i,1}, \ldots, \rho_{i,m_i}$ denote the set of repeating paths encoded in λ_1 that cross variables in $\mathbf{z}_i$. Finally, let Θ' denote the paths in σ_1' that connect r-indices with l-indices and Θ denote the paths in σ_1 that connect l-indices with r-indices. Let $\Theta(i) \subseteq \Theta$ and $\Theta'(i) \subseteq \Theta'$ be paths that cross only variables in $\mathbf{z}_i$. We define $p = \max\left\{\lceil\frac{|\sigma_1|}{|\lambda_1|}\rceil, \lceil\frac{|\sigma_1'|}{|\lambda_1|}\rceil\right\}$. Let $\rho_{i,1} \prec \cdots \prec \rho_{i,m_i}$ be the order in which subpaths $\rho_{i,1}, \ldots, \rho_{i,m_i}$ are visited in the path encoded in $\sigma_1.\lambda_1.\sigma_1'$. Then, by construction of zigzag automata, $\rho_{i,j}$ is forward if and only if $\rho_{i,j+1}$ is backward. Figures 20(a) and (b) illustrate these definitions.

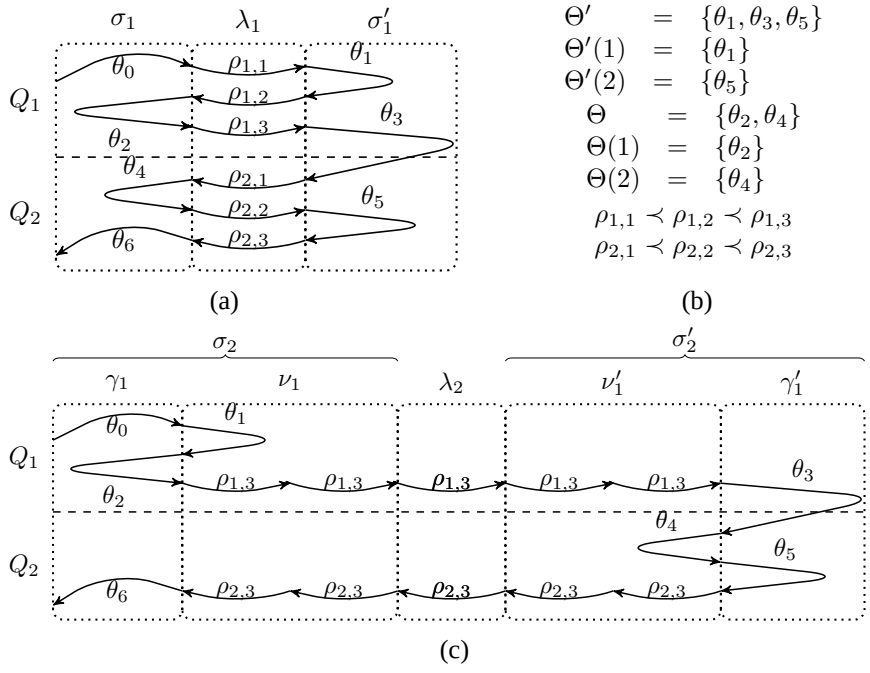


Figure 20: Illustration of a construction of a path scheme with simple cycle.

Given an equivalence class $\mathbf{z}_i$, $1 \leq i \leq m$, we first give a construction ensuring that there is at most one repeating path in λ_2 that crosses variables in $\mathbf{z}_i$.

We next build $\gamma, \nu, \lambda_2, \nu', \gamma'$, which are initialized as follows: $\gamma = \sigma_1$, $\nu = (\lambda_1)^p$, $\lambda_2 = \lambda_1$, $\nu' = (\lambda_1)^p$, $\gamma' = \sigma'_1$. Further, we erase all paths from ν, λ_2, ν' that cross some variable in $\mathbf{z}_i$. We finish construction of $\gamma, \nu, \lambda_2, \nu', \gamma'$ for one of the following cases (note that cases 1 and 2 (3 and 4) are symmetrical):

1. $\rho_{i,1}$ **is fw and** m_i **is even.** Erase $\Theta(i)$ from γ' and add it to ν.
2. $\rho_{i,1}$ **is bw and** m_i **is even.** Erase $\Theta'(i)$ from γ and add it to ν'.
3. $\rho_{i,1}$ **is fw and** m_i **is odd.** Do the actions for Case 1. Further, add ρ_{i,m_i} to λ_2 and add $(\rho_{i,m_i})^p$ to both ν and ν'.
4. $\rho_{i,1}$ **is bw and** m_i **is odd.** Do the actions for Case 2. Further, add ρ_{i,m_i} to λ_2 and add $(\rho_{i,m_i})^p$ to both ν and ν'.

The construction is finished by building $\sigma_2 = \gamma.\lambda_1^{b_1}\nu$ and $\sigma'_2 = \nu'.\gamma'$. Figure 20(c) illustrates the construction: Case 3 applies for the equivalence class $\mathbf{z}_1$, while the Case 4 applies for $\mathbf{z}_2$.

We prove several properties on weights of $\sigma_2, \lambda_2, \sigma'_2$ for case 3 (proofs for cases 1, 2, and 4 are similar). Let us define

$$W = \sum_{\theta \in \Theta(i)} w(\theta) \qquad V = \sum_{1 \leq j < m_i} w(\rho_{i,j})$$

Following equalities follow from the construction:

$$|\sigma_2.\sigma'_2| - |\sigma_1.\sigma'_1| = |\lambda_1^{b_1}| + |\nu| + |\nu'| = |\lambda_1| \cdot (b_1 + 2p)$$

$$\begin{array}{lcllcl} w(\gamma) & = & w(\sigma_1) & w(\nu') & = & p \cdot (w(\lambda_1) - V) \\ w(\nu) & = & p \cdot (w(\lambda_1) - V) + W & w(\gamma') & = & w(\sigma'_1) - W \end{array}$$

Then,

$$\begin{array}{rcl} w(\sigma_2.\sigma'_2) = w(\gamma.\lambda_1^{b_1}.\nu.\nu'.\gamma') & = & w(\sigma_1) + (b_1 + 2p) \cdot w(\lambda_1) + w(\sigma'_1) - 2pV, \\ w(\sigma_1.\lambda_1^{b_1+2p}.\sigma'_1) & = & w(\sigma_1) + (b_1 + 2p) \cdot w(\lambda_1) + w(\sigma'_1). \end{array}$$

For Case 3, m_i is odd. Since $\rho_{i,j}$ is fw and $\rho_{i,j+1}$ is bw for odd $j < m_i$, by Lemma 18, $w(\rho_j) + w(\rho_{j+1}) \geq 0$ and thus $V \geq 0$. By construction, $w(\lambda_2) = w(\lambda_1) - V$. Since $V > 0$ would imply that $w(\lambda_2) < w(\lambda_1)$ and thus, that $\sigma_1.\lambda_1^{b_1+k}.\sigma'_1$ is not minimal for all $k \geq 0$, we infer that $V = 0$ and therefore, $w(\lambda_2) = w(\lambda_1)$ and

$$w(\sigma_2.\sigma'_2) = w(\sigma_1.\lambda_1^{b_1+2p}.\sigma'_1).$$

Note that the above construction of σ_2, λ_2, and σ'_2 can be easily extended to deal with all equivalence classes of $\sim$ at once. Then, λ_2 is simple and the above equality still hold. Note that if the construction steps for Case 3 or Case 4 generate a path (encoded in ν or ν') with cycles, we erase all the cycles. Their weights must be non-negative, since we consider $*$-consistent relations. They also cannot be strictly positive since then $\sigma_1.\lambda_1.\sigma'_1$ would not be minimal for all $k \geq b_1 + 2p$, contradiction. Thus erasing

them changes weight of neither σ_2 nor σ_2' and hence the above equality still hold. Let us define $b_2' = 0$, $b_1' = b_1 + 2p$, $\rho_2' = \sigma_2.\lambda_2^{b_2'}\sigma_2'$, $\rho_1' = \sigma_1.\lambda_1^{b_1'}\sigma_1'$ and observe that

$$\begin{array}{rcl} L = |\rho_2'| - |\rho_1| & = & (|\sigma_2.\sigma_2'| - |\sigma_1.\sigma_1'|) + |\lambda_2^{b_2'}| - |\lambda_1^{b_1}| \\ & = & |\lambda_1| \cdot (b_1 + 2p) + 0 - |\lambda_1| \cdot b_1 \\ & = & |\lambda_1| \cdot (b_1 + 2p - b_1) = |\lambda_1| \cdot 2p \end{array}$$

and thus, $|\lambda_1| = |\lambda_2|$ divides L. Next, we apply Lemma 19 which guarantees existence of a path scheme $\theta_3 = \sigma_3.\lambda_2^*.\sigma_3'$ and a path $\rho_3 = \sigma_3.\lambda_2^{b_3}.\sigma_3'$, for some $b_3 \geq 0$, such that $w(\rho_3) = w(\rho_1)$ and $|\rho_3| = |\rho_1|$. The existence of $c \mid \frac{\mathrm{lcm}\ 1,\ldots,\|V\|}{|\lambda_2|}$ such that $\sigma_3.\lambda_2^{b_3+kc}.\sigma_3'$ is minimal for all $k \geq 0$ follows from the proof of Lemma 3, since we can choose $\pi_i.\lambda_i^*.\pi_i' = \theta_3$. □

8.3.5. Basic Path Schemes with Cycles Bounded by $\mathrm{lcm}(1, \ldots, N)$

This section refines the statement of Lemma 3 by proving that the length of the cycle λ from each basic path scheme divides $\mathrm{lcm}(1, \ldots, N)$. This fact is essential in proving the single exponential bound on the period of octagonal relations. We begin with a lemma that is later used to deal with the problem of synchronization of connecting paths which was discussed earlier. Recall that for a path $\rho : x_{i_0}^{(\ell_0)} \to \ldots \to x_{i_m}^{(\ell_m)}$ in $\mathcal{G}_R^\omega$ such that $i_0 = i_m$ and $\ell_m \neq \ell_0$, we defined its left (right) extent relative to $x_{i_0}^{(\ell_0)}$ as

$$\begin{array}{ll} L_\rho = |\min\{\ell_k - \ell_0 \mid 0 \leq k \leq m\}| & \overline{L}_\rho = \left\lceil \frac{L_\rho}{\|\rho\|} \right\rceil \\ R_\rho = |\max\{\ell_k - \ell_0 \mid 0 \leq k \leq m\}| & \overline{R}_\rho = \left\lceil \frac{R_\rho}{\|\rho\|} \right\rceil \end{array}$$

Lemma 21. *Let $\tau' = \mu^r.\xi.\nu^s.\zeta.\mu^t$ be a connecting path such that*

$$r = r_1 + r_2 \qquad s = s_1 + \cdots + s_5 \qquad t = t_1 + t_2, \text{ where}$$

$$\begin{array}{rclcrcl} s_1 & \geq & L_\nu + R_\mu + N + 1 & & t_1 & \geq & L_\mu + R_\nu + N + 1 \\ s_5 & \geq & R_\nu + L_\mu + N + 1 & & r_2 & \geq & R_\mu + L_\nu + N + 1 \end{array}$$

$$\begin{array}{rclcrcl} s_2 & \geq & \overline{R}_\nu - 1 & & & & t_2 \geq 1 + \overline{R}_\nu \\ s_4 & \geq & \overline{L}_\nu & \qquad s_3 \geq 2N \qquad & & & r_1 \geq 1 + \overline{L}_\nu \end{array}$$

Let τ be a path built by erasing all cycles from τ' and let

- *λ_μ be a zigzag cycle that encodes μ,*
- *λ_ν be a zigzag cycle that encodes ν,*
- *λ_τ be a zigzag cycle that encodes τ.*

Then, λ_τ can be written as $\lambda_\tau = \lambda_\mu.\pi_1.{\lambda_\nu'}^{s_3-2N}.\pi_2.\lambda_\mu$ where π_1, π_2 are some zigzag paths, $|\lambda_\nu| = |\lambda_\nu'|$, $w(\lambda_\nu) = w(\lambda_\nu')$, and $|\lambda_\mu.\pi_1| = \|\mu^{r_1+r_2}\| + \|\nu^{s_1+s_2}\| + N$.

Proof: Let us denote the subpaths of τ' as

$$\begin{array}{rcccccccccccccccccccc} & & \alpha_1 & & \alpha_2 & & \alpha_3 & & \alpha_4 & & \alpha_5 & & \alpha_6 & & \alpha_7 & & \alpha_8 & & \alpha_9 & & \alpha_{10} & & \alpha_{11} \\ & & || & & || & & || & & || & & || & & || & & || & & || & & || & & || \\ \tau' & = & \mu^{r_1} & . & \mu^{r_2} & . & \xi & . & \nu^{s_1} & . & \nu^{s_2} & . & \nu^{s_3} & . & \nu^{s_4} & . & \nu^{s_5} & . & \zeta & . & \mu^{t_1} & . & \mu^{t_2} \end{array}$$

First, we show that the constraint on s_1 ensures that subpaths $\alpha_1.\alpha_2.\alpha_3$ and α_5 do not share a node. Let $M_1 = \|\alpha_1.\alpha_2\|$ and observe that

$$vars(\alpha_1.\alpha_2) \subseteq \bigcup_{k=-L_\mu}^{M_1+R_\mu} \mathbf{x}^{(k)}, \qquad vars(\alpha_3) \subseteq \bigcup_{k=M_1-N}^{M_1+N} \mathbf{x}^{(k)}.$$

The property on $vars(\alpha_1.\alpha_2)$ holds since $L_{\rho^k} \leq L_\rho$ and $R_{\rho^k} \leq R_\rho$ for each path ρ and $k \geq 1$. The property on $vars(\alpha_3)$ follows from the fact that ξ is essential. Hence,

$$vars(\alpha_1.\alpha_2.\alpha_3) \subseteq \bigcup_{k=\min(-L_\mu,-N)}^{M_1+\max(R_\mu,N)} \mathbf{x}^{(k)} \subseteq \bigcup_{k=\min(-L_\mu,-N)}^{M_1+R_\mu+N} \mathbf{x}^{(k)}.$$

Similarly, letting $M_2 = M_1 + \|\alpha_3.\alpha_4\|$, we observe that

$$vars(\alpha_5) \subseteq \bigcup_{k=M_2-L_\nu}^{M_2+\|\alpha_5\|+R_\nu} \mathbf{x}^{(k)}.$$

We can now infer a condition that guarantees that subpaths $\alpha_1.\alpha_2.\alpha_3$ and α_5 do not share a node:

$$\begin{array}{rcll} M_1 + R_\mu + N & < & M_2 - L_\nu & \\ R_\mu + N & < & \|\alpha_3.\alpha_4\| - L_\nu & \\ \|\alpha_4\| & > & L_\nu + R_\mu + N - \|\alpha_3\| & \\ \|\alpha_4\| & \geq & L_\nu + R_\mu + N + 1 & \text{(sufficient, since } \|\alpha_3\| \geq 0\text{)} \\ s_1 & = & L_\nu + R_\mu + N + 1 & \text{(sufficient, since } \|\alpha_4\| = \|\nu\| \cdot s_1\text{)} \end{array}$$

Similarly, one infers constraints on s_5, t_1, and r_2. These constrains guarantee that all sharings of nodes (in other words, cycles) may occur only in $\alpha_2, \alpha_4, \alpha_8$, or α_{10} and thus, that no cycle appears in $\alpha_5.\alpha_6.\alpha_7$, α_1 and α_{11}.

Next, we prove an auxiliary statement that λ_τ can be written as $\lambda_\tau = \lambda_\mu.\pi_1.\lambda_2^{s_3}.\pi_2.\lambda_\mu$ where π_1, π_2 are some zigzag paths, and $|\lambda_\mu.\pi_1| = \|\mu^{r_1+r_2}.\xi\nu^{s_1+s_2}\|$. Figures 21 and 22 illustrate the proof.

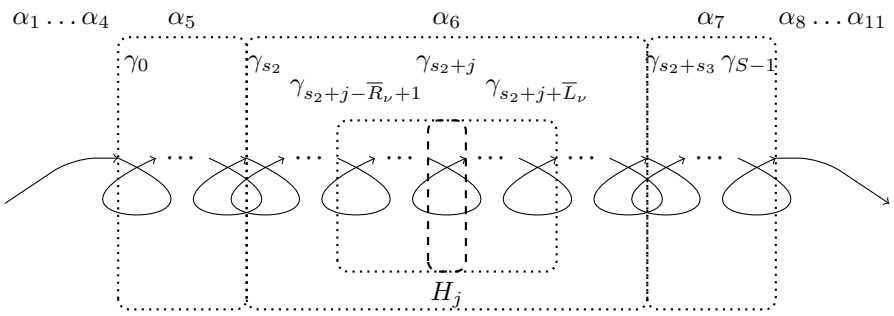
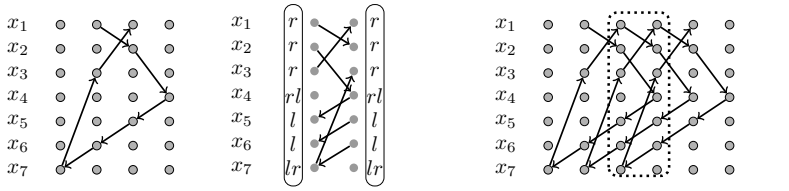


Figure 21: $\lambda_\nu^{s_3}$ as a subpath of λ_τ.



(a) path ν (b) zigzag cycle λ_ν (c) $H_j, \gamma_{s_2+j-1}, \gamma_{s_2+j}, \gamma_{s_2+j+1}$

Figure 22: Obtaining λ_ν by iterating ν. $\overline{L}_\nu = 1, \overline{R}_\nu = 2$.

Let $G_0 G_1 \ldots G_{|\lambda_\tau|-1}$ be the labeling of λ_τ and let $\alpha_5.\alpha_6.\alpha_7 = \gamma_0.\gamma_1 \ldots \gamma_{S-1}$, where $S = s_2 + s_3 + s_4$ and $\nu = \gamma_0 = \gamma_1 = \cdots = \gamma_{S-1}$. For each $0 \leq j < s_3$, define $K_j = \|\alpha_1 \ldots \alpha_5.\nu^j\|$, $H_j = G_{K_j+1} \ldots G_{K_j+\|\nu\|}$ and observe that for each $-\overline{R}_\nu < k \leq \overline{L}_\nu$, γ_{s_2+j+k} contributes[7] to H_j with $\Theta_{\nu,k}$. Thus, H_j consists of zigzag-segments

$$\bigcup_{j=-\overline{L}_\nu+1}^{\overline{R}_\nu} \Theta_{\nu,j}$$

which are clearly zigzag-segments of λ_ν. Thus, H_j is the labeling of λ_ν for each $0 \leq j < s_3$ and consequently, λ_τ can be decomposed into $\sigma_1.\lambda_\nu^{s_3}.\sigma_2$ for some paths σ_1, σ_2. Moreover, since $K_0 = \|\alpha_1 \ldots \alpha_5\| = \|\mu^{r_1+r_2}.\xi\nu^{s_1+s_2}\|$, then

$$|\sigma_1| = K_0 = \|\mu^{r_1+r_2}.\xi\nu^{s_1+s_2}\|.$$

By a similar argument, one can show that λ_τ can be decomposed into $\lambda_\mu.\sigma_3.\lambda_\mu$ for some path σ_3, by viewing the path $\tau' = \alpha_1 \ldots \alpha_{11}$ shifted as $\alpha_6 \ldots \alpha_{11}\alpha_1 \ldots \alpha_5$. Combining decompositions $\sigma_1.\lambda_\nu^{s_3}.\sigma_2$ and $\lambda_\tau = \lambda_\mu.\sigma_3.\lambda_\mu$, we obtain the required decomposition $\lambda_\tau = \lambda_\mu.\pi_1.\lambda_2^{s_3}.\pi_2.\lambda_\mu$ for some π_1, π_2, where

$$|\lambda_\mu.\pi_1| = |\sigma_1| = \|\mu^{r_1+r_2}.\xi\nu^{s_1+s_2}\|.$$

[7]Hence the bounds on s_2 and s_4: $s_2 \geq \overline{R}_\nu - 1$, $s_4 \geq \overline{L}_\nu$.

Finally, we prove that λ_τ can be written as $\lambda_\tau = \lambda_\mu.\pi_1.{\lambda'_\nu}^{s_3-2N}.\pi_2.\lambda_\mu$ where π_1, π_2 are some zigzag paths, $|\lambda_\nu| = |\lambda'_\nu|$, $w(\lambda_\nu) = w(\lambda'_\nu)$, and $|\lambda_\mu.\pi_1| = \|\mu^{r_1+r_2}\| + \|\nu^{s_1+s_2}\| + N$. Let us define

$$D_0 = \|\mu^{r_1+r_2}\|+\|\nu^{s_1+s_2}\|, \;\; D_1 = \|\mu^{r_1+r_2}.\xi.\nu^{s_1+s_2}\|, \;\; D_2 = \|\mu^{r_1+r_2}\|+\|\nu^{s_1+s_2}\|+N.$$

Since ξ is essential, $D_2 \geq D_1$ and $D_0 - N \leq D_1 \leq D_0 + N$. Thus, by noticing that $D_2 = D_0 + N$, we infer that $0 \leq D_2 - D_1 \leq 2N$. Clearly, there exists $0 \leq d < \|\nu\|$ and $k \geq 0$ such that $D_2 + d = D_1 + k\|\nu\|$. Thus, $k = \lceil \frac{D_2 - D_1}{\|\nu\|} \rceil$. Combining this with $0 \leq D_2 - D_1 \leq 2N$, we establish a bound $k \leq 2N$.

Next, observe that a decomposition of λ_τ into $\lambda_\mu.\pi_3.\lambda_\nu^{s_3-k}.\pi_4.\lambda_\mu$, where $|\lambda_\mu.\pi_3| = D_2 + d = D_1 + k\|\nu\|$, is possible by a similar argument as previously. The only difference is that j ranges over $k \leq j < s_3$ instead of $0 \leq j < s_3$. Thus, we need to guarantee that $s_3 - k \geq 0$. This can be achieved by requiring that $s_3 \geq 2N$, since the maximal value of k is $k = 2N$. This gives the stricter condition on s_3 in this lemma that guarantees the decomposition into $\lambda_\mu.\pi_3.\lambda_\nu^{s_3-2N}.\pi_4.\lambda_\mu$, where $|\lambda_\mu.\pi_3| = D_2 + d = D_1 + k\|\nu\|$.

Let $G_1 \dots G_n$ be the labeling of λ_ν. The decomposition in the previous paragraph implies that the label of $(\lambda_\tau)_{D_2+d\dots|\lambda_\tau|-1}$ has a prefix $(G_1 \dots G_n)^{s_3-2N}$. Further, the label of $(\lambda_\tau)_{D_1\dots|\lambda_\tau|-1}$ has a prefix $(G_1 \dots G_n)^{s_3}$. Since $D_2 + d - D_1 = k\|\nu\|$, it follows that the label of $(\lambda_\tau)_{D_1\dots D_2-1}$ is $(G_1 \dots G_n)^k$. Furthermore, we infer that the label of $(\lambda_\tau)_{D_2\dots|\lambda_\tau|-1}$ has a prefix $(G_{d+1} \dots G_n G_1 \dots G_d)^{s_3-2N}$. Since $G_1 \dots G_n$ is the labeling of the cycle $\lambda_\nu : q \xrightarrow{G_1\dots G_d} q' \xrightarrow{G_{d+1}\dots G_n} q$, there clearly exists a cycle $\lambda'_2 : q' \xrightarrow{G_{d+1}\dots G_n} q \xrightarrow{G_1\dots G_d} q'$. Thus, λ can be decomposed into $\lambda_\tau = \sigma_1.(\lambda'_\nu)^{s_3-2N}.\sigma_2$, where $|\sigma_1| = D_2 = \|\mu^{r_1+r_2}\| + \|\mu^{s_1+s_2}\| + N$. Clearly, $|\lambda_\nu| = |\lambda'_\nu|$ and $w(\lambda_\nu) = w(\lambda'_\nu)$. □

Informally, the following technical proposition states that the relative lengths of two repeating paths can be synchronized by iterating each repeating path with itself several times.

Proposition 22. *Let γ_1, γ_2 be repeating paths and let $c_1, c_2 \geq 1$. Then, there exists $c'_1 = c_1 \cdot k_1$, $c'_2 = c_2 \cdot k_2$ for some $k_1, k_2 \geq 1$ such that $\|\gamma_1^{c'_1}\| = \|\gamma_2^{c'_2}\|$.*

Proof: Let $L = \mathrm{lcm}(c_1 \cdot \|\gamma_1\|, c_2 \cdot \|\gamma_2\|)$, $c'_1 = \frac{L}{\|\gamma_1\|}$, $c'_2 = \frac{L}{\|\gamma_2\|}$. Since $c_1 \cdot \|\gamma_1\|$ divides L, then c_1 divides $\frac{L}{\|\gamma_1\|}$ too and thus, there exists $k_1 \geq 1$ such that $c_1 \cdot k = c'_1$. Similarly, there exists $k_2 \geq 1$ such that $c_2 \cdot k = c'_2$. Further, $\|\gamma_1^{c'_1}\| = \|\gamma_2^{c'_2}\| = L$, since $\|\gamma_1^{c'_1}\| = \|\gamma_1\| \cdot c'_1 = \|\gamma_1\| \cdot \frac{L}{\|\gamma_1\|} = L$ and similarly, $\|\gamma_2^{c'_2}\| = L$. □

We finally prove that we can, without loss of generality, consider basic path schemes with cycles that are simple and moreover, the length of which divides $\mathrm{lcm}(1, \dots, N)$. For the proof of the lemma, we need the notion of *optimal* cycle.

Definition 31. *a simple cycle λ is* optimal *if and only if each forward path encoded in λ is fw-optimal and each backward path encoded in λ is bw-optimal.*

Lemma 22. *Let $T_R = (Q, \Delta, w)$ be the common transition table of zigzag automata defined for a difference bounds relation $R(\mathbf{x}, \mathbf{x}')$, and $u, v \in Q$ be two control states. Then for every minimal weight path ρ form u to v, such that $|\rho| \geq \|R\| \cdot \|V\|^6$, there exists a path ρ' from u to v, such that $w(\rho) = w(\rho')$ and $|\rho| = |\rho'|$, and a basic path scheme $\theta = \sigma \cdot \lambda^* \cdot \sigma'$, such that λ is simple and $|\lambda|$ divides $\mathrm{lcm}(1, \ldots, N)$, $\rho' = \sigma \cdot \lambda^b \cdot \sigma'$, for some $b \geq 0$. Moreover, there exists $c \mid \frac{\mathrm{lcm}(1,\ldots,N)}{|\lambda|}$ such that $\sigma \cdot \lambda^{b+kc} \cdot \sigma'$ is a minimal weight path from u to v, for all $k \geq 0$.*

Proof: By Lemma 20, there exists a basic path scheme $\theta_1 = \sigma_1.\lambda_1^*.\sigma_1'$ where λ_1 is simple and a path $\rho_1 = \sigma_1.\lambda_1^{b_1}.\sigma_1'$, for some $b_1 \geq 0$, such that $w(\rho_1) = w(\rho)$ and $|\rho_1| = |\rho|$. In this proof, we assume that λ_1 encodes two forward paths μ_1, μ_2 and no backward path. The extension to arbitrary number of forward and backward paths is straightforward. Let μ_j be of the form $\mu_j : x_{i_j} \rightsquigarrow x_{i_j}$ for each $j \in \{1, 2\}$. and let us denote the equivalence class of x_{i_j} as $\mathbf{z}_j = [x_{i_j}]_\sim$.

Case 1: λ_1 is optimal. By Proposition 20, given μ_j, there exists $\nu_j \in P^c_\triangleright(\mathbf{z}_j)$ of the form $\nu_j : x_{k_j} \rightsquigarrow x_{k_j}$ and two essential paths $\xi_j : x_{i_j} \rightsquigarrow x_{k_j}$ and $\zeta_j : x_{k_j} \rightsquigarrow x_{i_j}$. For each $j \in \{1, 2\}$, we build a connecting path τ_j' as follows:

$$\begin{array}{rcccccccccccccccccccc}
 & \alpha_1 & & \alpha_2 & & \alpha_3 & & \alpha_4 & & \alpha_5 & & \alpha_6 & & \alpha_7 & & \alpha_8 & & \alpha_9 & & \alpha_{10} & & \alpha_{11} \\
 & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel \\
\tau_1' = & \mu_1^{r_1} & . & \mu_1^{r_2} & . & \xi_1 & . & \nu_1^{s_1} & . & \nu_1^{s_2} & . & \nu_1^{s_3} & . & \nu_1^{s_4} & . & \nu_1^{s_5} & . & \zeta_1 & . & \mu_1^{t_1} & . & \mu^{t_2} \\
\tau_2' = & \mu_2^{t_3} & . & \mu_2^{t_4} & . & \xi_2 & . & \nu_2^{w_1} & . & \nu_2^{w_2} & . & \nu_2^{w_3} & . & \nu_2^{w_4} & . & \nu_2^{w_5} & . & \zeta_2 & . & \mu_2^{t_1} & . & \mu^{t_2} \\
 & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel \\
 & \beta_1 & & \beta_2 & & \beta_3 & & \beta_4 & & \beta_5 & & \beta_6 & & \beta_7 & & \beta_8 & & \beta_9 & & \beta_{10} & & \beta_{11}
\end{array}$$

In addition to the conditions of Lemma 21, we require that $|\alpha_k| = |\beta_k|$ for all $k \in \{1, 2, 4, 5\}$. These additional constraints can be satisfied too, by Proposition 22. Clearly, the new coefficients still satisfy the conditions in Lemma 21. Finally, we define $s_3 = 2N + \frac{L}{\|\nu_1\|}$, $w_3 = 2N + \frac{L}{\|\nu_2\|}$, where $L = \mathrm{lcm}(\|\nu_1\|, \|\nu_2\|)$.

By Lemma 21, there exists a zigzag cycles λ_{τ_1} and λ_{τ_2} that encode τ_1 and τ_2 that were obtained by erasing all cycles in τ_1' and τ_2', respectively, and that can be decomposed into

$$\begin{array}{rcl}
\lambda_{\tau_1} & = & \lambda_{\mu_1}.\pi_1.{\lambda'_{\nu_1}}^{s_3-2N}.\pi_2.\lambda_{\mu_1} \\
\lambda_{\tau_2} & = & \lambda_{\mu_2}.\pi_3.{\lambda'_{\nu_2}}^{w_3-2N}.\pi_4.\lambda_{\mu_2}
\end{array}$$

where

$$|\lambda'_{\nu_j}| = |\lambda_{\nu_j}| \text{ and } w(\lambda'_{\nu_j}) = w(\lambda_{\nu_j}) \text{ and}$$

$$\|\lambda_{\mu_1}.\pi_1\| = \|\alpha_1.\alpha_2\| + \|\alpha_4.\alpha_5\| + N \text{ and } \|\beta_1.\beta_2\| + \|\beta_4.\beta_5\| + N = \|\lambda_{\mu_2}.\pi_3\|$$

Since $|\alpha_k| = |\beta_k|$ for all $k \in \{1, 2, 4, 5\}$, we infer that $\|\lambda_{\mu_1}.\pi_1\| = \|\lambda_{\mu_2}.\pi_3\|$.

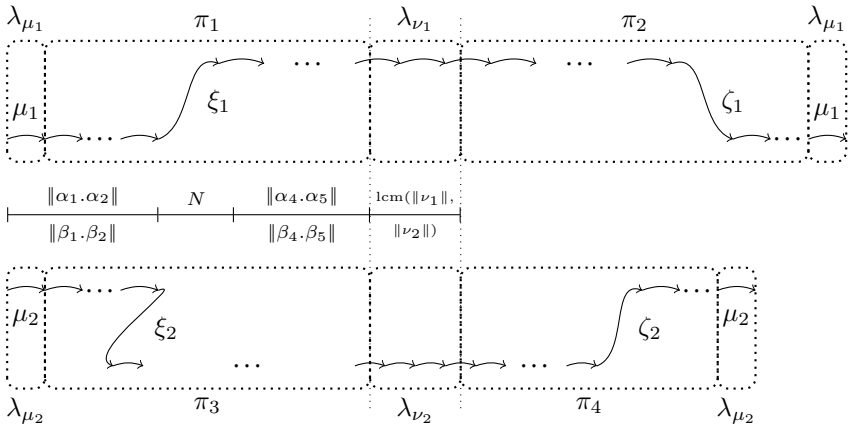


Figure 23: Synchronization of connecting paths τ_1 and τ_2.

Note that $|\lambda_{\tau_1}| = |\lambda_{\tau_2}|$ is not true in general. For this reason, we need to make an extra step. Let $M = \mathrm{lcm}(|\lambda_{\tau_1}|, |\lambda_{\tau_2}|)$, $m_1 = \frac{M}{|\lambda_{\tau_1}|}$, $m_2 = \frac{M}{|\lambda_{\tau_2}|}$. Since $|\lambda_{\tau_1}^{m_1}| = |\lambda_{\tau_2}^{m_2}|$, and since the paths in λ_{τ_1} and λ_{τ_2} use disjoint variables, we can build a cycle λ by gluing $\lambda_{\tau_1}^{m_1}$ with $\lambda_{\tau_2}^{m_2}$ and obtain $\lambda = \lambda_1.\pi_1'.\lambda_2'.\pi_2'.\lambda_1$, where

$$\lambda_1 = \begin{bmatrix} \lambda_{\mu_1} \\ \lambda_{\mu_2} \end{bmatrix} \pi_1' = \begin{bmatrix} \pi_1 \\ \pi_3 \end{bmatrix} \lambda_2' = \begin{bmatrix} {\lambda'_{\nu_1}}^{s_3-2N} \\ {\lambda'_{\nu_2}}^{w_3-2N} \end{bmatrix} \pi_2' = \begin{bmatrix} \pi_2.\lambda_{\mu_1}.(\lambda_{\tau_1})^{m_1-1} \\ \pi_4.\lambda_{\mu_2}.(\lambda_{\tau_2})^{m_2-1} \end{bmatrix}$$

Note that the construction of λ_2' is correct since

$$|{\lambda'_{\nu_1}}^{s_3-2N}| = |\lambda_{\nu_1}^{s_3-2N}| = \|\nu_1\|(s_3-2N) = \|\nu_1\|\left(\tfrac{L}{\|\nu_1\|} + 2N - 2N\right) = \mathrm{lcm}(\|\nu_1\|, \|\nu_2\|)$$

and similarly, $|{\lambda'_{\nu_2}}^{w_3-2N}| = \mathrm{lcm}(\|\nu_1\|, \|\nu_2\|)$. Thus, $|\lambda_2'| = \mathrm{lcm}(\|\nu_1\|, \|\nu_2\|)$. Note that λ_1 is optimal by assumption. Further, λ_2 encodes paths $\nu_1^{s_3}$ and $\nu_2^{w_3}$ which are optimal, by the fact that ν_1, ν_2 are optimal and by Proposition 19. Thus, λ_2 is optimal by construction. Since $\|\nu_1\|, \|\nu_1\| \leq N$, then $|\lambda_2|$ divides $\mathrm{lcm}(1, \ldots, N)$. By Proposition 20, $\nu_j \in \mathcal{F}(\mu_j)$ and $\mathcal{F}(\xi_j.\zeta_j) \subseteq \mathcal{F}(\mu_j)$. Thus, $\mathcal{F}(\tau_j) \subseteq \mathcal{F}(\mu_j)$. Since τ_j is forward, then $\overline{\overline{w}}(\tau_j) = C_{\triangleright}(\mathbf{z}_j)$, by Lemma 17. Consequently, τ_j is optimal. By Lemma 21, λ_{τ_j} encodes τ_j' that was obtained from τ_j' by erasing all its cycles. These cycles are non-negative, since R is $*$-consistent. Next suppose that at least one is strictly positive. Then,

$$\overline{\overline{w}}(\tau_j) < \overline{\overline{w}}(\tau_j') = \overline{\overline{w}}(\mu_j) = \overline{\overline{w}}(\nu_j) = C_{\triangleright}(\mathbf{z}_j).$$

However, by Lemma 17, $\overline{\overline{w}}(\tau_j) \geq C_{\triangleright}(\mathbf{z}_j)$, contradiction. Thus, $\overline{\overline{w}}(\tau_j) = C_{\triangleright}(\mathbf{z}_j)$ too. Consequently, $\lambda_1, \lambda_2, \lambda$ are optimal too, by Definition 31. By Proposition 21,

$$\overline{w}(\lambda_1) = \overline{w}(\lambda_2) = \overline{w}(\lambda) = C_{\triangleright}(\mathbf{z}_1) + C_{\triangleright}(\mathbf{z}_2).$$

We next construct a path scheme $\theta_2 = \sigma_2.{\lambda_2'}^*.\sigma_2'$, where

$$\sigma_2 = \sigma_1.\lambda_1^{b_1}.\lambda_1.\pi_1' \qquad \sigma_2' = \lambda_2'.\pi_2'.\lambda_1.\lambda^{(|\lambda_1|\cdot|\lambda_2'|-1)}.\sigma_1'$$

Next, letting $b'_2 = |\lambda_1|$, we construct $\rho'_2 = \sigma_2.{\lambda'_2}^{b'_2}.\sigma'_2$. Recalling that $\rho_1 = \sigma_1.\lambda_1^{b_1}.\sigma'_1$, we compute

$$\begin{aligned} D = |\rho'_2| - |\rho_1| &= |\lambda_1.\pi'_1.\lambda'_2.\pi'_2.\lambda_1.\lambda^{|\lambda_1|\cdot|\lambda'_2|-1}| + |{\lambda'_2}^{|\lambda_1|}| \\ &= |\lambda^{|\lambda_1|\cdot|\lambda'_2|}| + |\lambda_1| \cdot |\lambda'_2| \\ &= |\lambda_1| \cdot |\lambda'_2| \cdot (|\lambda| + 1) \end{aligned}$$

Letting $b'_1 = b_1 + \frac{D}{|\lambda_1|} + |\lambda'_2|$, we construct $\rho'_1 = \sigma_1.\lambda_1^{b'_1}.\sigma'_1$. Clearly, $|\rho'_1| = |\rho'_2|$. We infer that

$$\begin{aligned} w(\rho'_2) - w(\rho_1) &= |\lambda_1| \cdot |\lambda'_2| \cdot |\lambda| \cdot \overline{w}(\lambda) + |\lambda_1| \cdot |\lambda_2| \cdot \overline{w}(\lambda_2) \\ &= D \cdot (C_\triangleright(\mathbf{z}_1) + C_\triangleright(\mathbf{z}_2)) \\ w(\rho'_1) - w(\rho_1) &= |\lambda_1| \cdot (\tfrac{D}{|\lambda_1|} + |\lambda'_2|) \cdot \overline{w}(\lambda_1) \\ &= D \cdot (C_\triangleright(\mathbf{z}_1) + C_\triangleright(\mathbf{z}_2)) \end{aligned}$$

and thus, $w(\rho'_2) = w(\rho'_1)$. Clearly, $|\lambda_1|$ and $|\lambda'_2|$ divides D. We apply Lemma 19 which guarantees existence of a path scheme $\theta_3 = \sigma_3.{\lambda'_2}^*.\sigma'_3$ and a path $\rho_3 = \sigma_3.{\lambda'_2}^{b_3}.\sigma'_3$, for some $b_3 \geq 0$, such that $w(\rho_3) = w(\rho)$ and $|\rho_3| = |\rho|$. Thus, the lemma holds for $\theta = \theta_3$ and $\rho' = \rho_3$.

Case 2: λ_1 is not optimal. Since $\mu_1 \in P_\triangleright(\mathbf{z}_1)$, then $P^c_\triangleright(\mathbf{z}_1) \neq \emptyset$. Let choose $\nu_1 \in P^c_\triangleright(\mathbf{z}_1)$ and assume its form is $\mu_1 : x_{k_1} \rightsquigarrow x_{k_1}$. Further, let $\xi_1 : x_{i_1} \rightsquigarrow x_{k_1}$, $\zeta_1 : x_{k_1} \rightsquigarrow x_{i_1}$ be arbitrary essential paths. Similarly for μ_2, we construct ν_2, ξ_2, ζ_2. We construct $\lambda, \theta_2, \rho'_1, \rho'_2$ and compute D in the same way as in Case 1. Clearly, λ'_2 is optimal, λ_1 is not optimal and thus $\overline{w}(\lambda_1) > \overline{w}(\lambda'_2) = C_\triangleright(\mathbf{z}_1) + C_\triangleright(\mathbf{z}_2)$. Since $|\lambda_1|$ and $|\lambda'_2|$ divide $|\rho'_2| - |\rho_1|$, then $\sigma_1.\lambda_1^{b_1+ck}.\sigma'_2$ is not minimal for some $k \geq 0$. Contradiction with our assumption on $\theta_1 = \sigma_1.\lambda_1^*.\sigma'_1$.

We have proved that Case 2 is not possible, and that for Case 1, there exists a path $\rho_3 = \sigma_3.{\lambda'_2}^{b_3}.\sigma'_3$ where $w(\rho_3) = w(\rho)$, $|\rho_3| = |\rho|$, λ'_2 is optimal and $|\lambda'_2|$ divides $\mathrm{lcm}(1, \ldots, N)$. Let us denote $\rho' = \rho_3$, $\sigma = \sigma_3$, $\sigma' = \sigma'_3$, $\lambda = \lambda'_2$, $b = b_3$. It remains to prove that there exists $c \mid \frac{\mathrm{lcm}(1,\ldots,N)}{|\lambda'_2|}$ such that $\sigma.\lambda^{b+kc}.\sigma'$ is a minimal path from u to v for all $k \geq 0$. The proof is almost identical to that of Lemma 7. The only difference is that we can now consider only basic path schemes $\sigma.\lambda^*.\sigma'$ where λ is simple and $|\lambda|$ divides $\mathrm{lcm}(1, \ldots, N)$. Thus, the proof can use $\mathrm{lcm}(1, \ldots, N)$ instead of $\mathrm{lcm}(1, \ldots, \|V\| - 1)$ everywhere. This in turn implies the existence of $c \mid \mathrm{lcm}(1, \ldots, N)$ such that $w(\sigma_i.\lambda^{kc}.\sigma'_i)$ is minimal for all $k \geq 0$. □

The single exponential bound on the period of difference bounds relations follows easily from Lemma 22.

Corollary 7. *Let $\mathbf{x} = \{x_1, \ldots, x_N\}$ be a set of variables. Given a difference bounds relation $R(\mathbf{x}, \mathbf{x}')$, the period of $R(\mathbf{x}, \mathbf{x}')$ is bounded by $2^{\mathcal{O}(N)}$.*

Proof: Let $T_R = (Q, \Delta, w)$ be the common transition table of zigzag automata defined for a difference bounds relation $R(\mathbf{x}, \mathbf{x}')$ and let c be the period of T_R. By Lemma 22, $c \mid lcm(1, \ldots, N)$. Applying Lemma 8, it follows that c is bounded by $2^{\mathcal{O}(N)}$. □

We finally summarize the complexity results on difference bounds relations.

Theorem 13. *Let $\mathbf{x} = \{x_1, \ldots, x_N\}$ be a set of variables. Given a difference bounds relation $R(\mathbf{x}, \mathbf{x}')$, its period is bounded by $2^{O(N)}$ and its prefix is bounded by $\|R\| \cdot 2^{O(N)}$.*

Proof: Follows from Corollary 5 and Corollary 7. □

8.4. Octagonal Relations

Let $R(\mathbf{x}, \mathbf{x}')$ be an octagonal relation and $\overline{R}(\mathbf{y}, \mathbf{y}')$ be its difference bounds representation. Using the results on bounds on the prefix (Corollary 5) and the period (Corollary 7) of $\overline{R}(\mathbf{y}, \mathbf{y}')$, we infer, using Lemma 12, the bounds on the prefix and period of the relations $R(\mathbf{x}, \mathbf{x}')$ itself.

Theorem 14. *Let $\mathbf{x} = \{x_1, \ldots, x_N\}$ be a set of variables. Given a relation $R(\mathbf{x}, \mathbf{x}') \in \mathcal{R}_{oct}$, its period is bounded by $2^{O(N)}$ and its prefix is bounded by $\|R\|^2 \cdot 2^{O(N)}$.*

Proof: Let $\overline{R}(\mathbf{y}, \mathbf{y}')$ be the difference bounds representation of $R(\mathbf{x}, \mathbf{x}')$ and let $\mathcal{G}_{\overline{R}} = \langle Q, \Delta, w \rangle$ be the zigzag automaton of $\overline{R}(\mathbf{y}, \mathbf{y}')$. It follows immediately from Lemma 22 that $\mathcal{G}_{\overline{R}}$ has prefix $b = \mu(\mathcal{G}_{\overline{R}}) \cdot \|Q\|^6 = \|\overline{R}\| \cdot 5^{12N}$ and period $c = \mathrm{lcm}(1, \ldots, 2N)$. Consequently, the prefix and period of $\{M^*_{\overline{R}^m}\}_{m \geq 0}$ and of $\overline{R}$ are b and c as well, respectively.

The prefix and the period of R are defined as the prefix and period of the sequence $\{\sigma(R^m)\}_{m \geq 0}$, by Definition 4. By definition of σ for octagonal relations given in Section 7.5, $\{\sigma(R^m)\}_{m \geq 0} = \{M^t_{\overline{R}^m}\}_{m \geq 0}$. By Theorem 4, $M^t_{\overline{R}^m} = M^{\bar{t}}_{\overline{R}^m}$ for all $m \geq 0$ and

$$(M^t_{\overline{R}^m})_{ij} = \min \left\{ (M^*_{\overline{R}^m})_{ij}, \left\lfloor \frac{(M^*_{\overline{R}^m})_{i\bar{i}}}{2} \right\rfloor + \left\lfloor \frac{(M^*_{\overline{R}^m})_{\bar{j}j}}{2} \right\rfloor \right\}$$

for all $m \geq 0$ and for all $1 \leq i, j \leq 4N$.

We next prove the asymptotic bound on period of R. If R is $*$-consistent, its period is twice the period of $\overline{R}$, by Lemma 12. Thus the period of R is bounded by $c = 2 \cdot \mathrm{lcm}(1, \ldots, 2N)$ and consequently, it is asymptotically bounded by $2^{O(N)}$, by Lemma 8. If R is not $*$-consistent, its period is 1 and the same asymptotic bound applies.

Next, we prove the asymptotic bound on the prefix of a $*$-consistent octagonal relation R. Let us define:

$$\{s_m\}_{m \geq 0} = \left\{ (M^*_{\overline{R}^k})_{i,j} \right\}_{m \geq 0} \qquad \{t_m\}_{m \geq 0} = \left\{ \left\lfloor \frac{(M^*_{\overline{R}^k})_{i,\bar{i}}}{2} \right\rfloor + \left\lfloor \frac{(M^*_{\overline{R}^k})_{\bar{j},j}}{2} \right\rfloor \right\}_{m \geq 0}$$

By Lemma 12, the periodic sequence $\{t_m\}_{m \geq 0}$ has prefix b and period $c' = 2c$. The sequence $\{s_m\}_{m \geq 0}$ has prefix b and period c, but we can without loss of generality assume that its period is $c' = 2c$. By Lemma 12, the sequence $\{\min(s_m, t_m)\}_{m \geq 0}$ has period c and prefix defined as $b' = b + \max_{i=0}^{c-1} K_i c'$ where

$$\begin{aligned}
K_i &= \left\lceil \tfrac{s_{b+i} - t_{b+i}}{\lambda_i^{(t)} - \lambda_i^{(s)}} \right\rceil && \text{if } \lambda_i^{(s)} < \lambda_i^{(t)} \text{ and } t_{b+i} < s_{b+i}, \\
K_i &= \left\lceil \tfrac{t_{b+i} - s_{b+i}}{\lambda_i^{(s)} - \lambda_i^{(t)}} \right\rceil && \text{if } \lambda_i^{(t)} < \lambda_i^{(s)} \text{ and } s_{b+i} < t_{b+i}, \\
K_i &= 0 && \text{otherwise.}
\end{aligned}$$

Observe that

$$\begin{array}{rcl} s_b & \geq & -b \cdot \|\overline{R}\|, \\ t_b & \leq & \max\{(M^*_{\overline{R}^b})_{i,\bar{i}}, (M^*_{\overline{R}^b})_{\bar{j},j}\} \leq b \cdot \|\overline{R}\|. \end{array}$$

Thus, if $\lambda_i^{(s)} > \lambda_i^{(t)}$ and $t_{b+i} > s_{b+i}$, then

$$K_i = \left\lceil \frac{t_{b+i} - s_{b+i}}{\lambda_i^{(s)} - \lambda_i^{(t)}} \right\rceil \leq t_{b+i} - s_{b+i} \leq 2 \cdot b \cdot \|\overline{R}\|.$$

Similarly, we infer that $K_i \leq 2 \cdot b \cdot \|\overline{R}\|$ if $\lambda_i^{(s)} < \lambda_i^{(t)}$ and $t_{b+i} < s_{b+i}$. Hence, $b' = b + 2 \cdot b \cdot \|\overline{R}\| \cdot c'$ is the prefix of $\{\min(s_m, t_m)\}_{m \geq 0}$ and thus of R. The asymptotic bound $\|R\|^2 \cdot 2^{\mathcal{O}(N)}$ on b' follows.

Finally, we prove the asymptotic bound on the prefix of R that is not $*$-consistent. Let b and c be the prefix and period of $\{M^*_{\overline{R}^m}\}_{m \geq 0}$ as inferred previously. By Theorem 3, either $M^*_{\overline{R}^\ell}$ is inconsistent or $\left\lfloor \frac{(M^*_{\overline{R}^\ell})_{i,\bar{i}}}{2} \right\rfloor + \left\lfloor \frac{(M^*_{\overline{R}^\ell})_{\bar{i},i}}{2} \right\rfloor < 0$ for some $\ell \geq 0$, $1 \leq i \leq 4N$. For the former case, ℓ (and thus the prefix of R) is bounded by $\|R\| \cdot 2^{\mathcal{O}(N)}$, by Corollary 5. Now consider the latter case. Let us denote

$$\{s_m\}_{m \geq 0} = \left\{ \left\lfloor \frac{(M^*_{\overline{R}^m})_{i,\bar{i}}}{2} \right\rfloor + \left\lfloor \frac{(M^*_{\overline{R}^m})_{\bar{i},i}}{2} \right\rfloor \right\}_{m \geq 0}$$

and let $\ell \geq 0$ and $1 \leq i \leq 4N$ be such that $s_\ell < 0$. If $\ell \leq b$, we immediately get the required asymptotic bound. If $\ell > b$, then by Lemma 22, there exist path schemes in the zigzag automaton $\sigma_1.\lambda_1^*.\sigma_1'$ and $\sigma_2.\lambda_2^*.\sigma_2'$ such that $(M^*_{\overline{R}^\ell})_{i,\bar{i}} = w(\sigma_1.\lambda_1^{b_1}.\sigma_1')$ for some $b_1 \geq 0$ and $(M^*_{\overline{R}^\ell})_{\bar{i},i} = w(\sigma_2.\lambda_2^{b_2}.\sigma_2')$ for some $b_2 \geq 0$ and moreover, letting $c_1 = \frac{c}{|\lambda_1|}$ and $c_2 = \frac{c}{|\lambda_2|}$, the paths $\sigma_1.\lambda_1^{b_1 + kc_1}.\sigma_1'$ and $\sigma_2.\lambda_2^{b_2 + kc_2}.\sigma_2'$ are minimal for all $k \geq 0$. By Lemma 12, the sequence $\{s_m\}_{m \geq 0}$ has prefix b and period $2c$. Moreover, its rate is $w(\lambda_1^{c_1}) + w(\lambda_2^{c_2})$. Clearly, $w(\lambda_1^{c_1}) + w(\lambda_2^{c_2}) < 0$, since otherwise $s_\ell < 0$ would not be possible. Observe that

$$s_b \leq \max\left\{ (M^*_{\overline{R}^b})_{i,\bar{i}}, (M^*_{\overline{R}^b})_{\bar{i},i} \right\} \leq b \cdot \|\overline{R}\|.$$

Then,

$$\ell \leq b \cdot \|\overline{R}\| + \left\lceil \frac{b \cdot \|\overline{R}\|}{-\left(w(\lambda_1^{c_1}) + w(\lambda_2^{c_2})\right)} \right\rceil \cdot c \leq b \cdot \|\overline{R}\| \cdot c.$$

Thus, ℓ and consequently the prefix of R are asymptotically bounded by $\|R\|^2 \cdot 2^{O(n)}$. □

8.5. Finite Monoid Affine Relations

An affine relation $R \in \mathbb{Z}^N \times \mathbb{Z}^N$ is defined by a linear arithmetic constraint of the form $\mathbf{x}' = A\mathbf{x} + \mathbf{b}$ where $A \in \mathbb{Z}^{N \times N}$ is a square matrix, and $\mathbf{b} \in \mathbb{Z}^N$ is a column vector. The relation is said to have the finite monoid property if the set $\{A^0, A^1, \ldots\}$ of matrix powers of A is finite. The cardinality of this set of called the *monoid size of* R, and denoted by $[R]$. Finite monoid affine relations are periodic, and the prefix b and period c of a relation R are such that $b + c = [R]$. In this section, we show that the

monoid size of a finite monoid relation is bounded by $2^{\mathcal{O}(N^{\log_{10} 11})}$, in other words, it is simply exponential in the number of variables. The developments in this section are closely related to decidability of the finite monoid property as mentioned in Theorem 5.

If R has the finite monoid property, then $[R]$ is the smallest integer p from the above theorem. This is because, due to the two conditions of Theorem 5, for every $k > 0$, we have that $A^{kp} = A^p$. Moreover, if p is the minimal integer satisfying these conditions, all powers $A^0, A^1, \ldots, A^{p-1}$ are pairwise distinct.

In the rest of this section, we give an upper bound for the smallest integer p that satisfies the conditions of Theorem 5. Notice first that every eigenvalue of A^p is of the form λ^p where λ is an eigenvalue of A. Since, by the first condition, λ^p is either zero or one, the only non-zero roots of the characteristic polynomial of A must be roots of the unity. But then $P_A(x)$ is a product of x^k, for some $k < N$, and several cyclotomic polynomials, call them $F_{i_1}, \ldots, F_{i_m}$. Clearly, the degrees of these polynomials are smaller than the degree of P_A, which, in turn, is smaller or equal to N. Let $i_0 = lcm(i_1, \ldots, i_m)$. Then every root λ of P_A has the property $\lambda^{i_0} = 1$, i.e. the first condition from Theorem 5 is met for $p = i_0^\ell$, for any integer $\ell > 0$. Moreover, this condition does not hold for any $0 < q < i_0$, or $i_0^\ell < q < i_0^{\ell+1}$, for all $\ell > 0$. But then the second condition of Theorem 5, if it holds for some p which is a multiple of i_0, it must hold also for $p = i_0$.

The only remaining question is how big $i_1, \ldots, i_m$ are. The idea is that we do not need to consider cyclotomic polynomials of degree higher than the degree of P_A, which is turn is at most N. A tight bound on the degree of a cyclotomic polynomial is given by the following theorem:

Theorem 15. *For every two integers $n > 0$ and $d \geq 0$, such that $n > 210\left(\frac{d}{48}\right)^{\log_{10} 11}$, the degree of $F_n(x)$ is higher than d.*

Proof: See Theorem 8.46 in [9]. □

Since, by Theorem 15, we have $0 < i_1, \ldots, i_m < 210\left(\frac{N}{48}\right)^{\log_{10} 11}$, it follows by Lemma 8 that $lcm(i_1, \ldots, i_m)$, and implicitly, the minimal integer p satisfying Theorem 5, is bounded by $2^{\mathcal{O}(N^{\log_{10} 11})}$. This gives the bound on the size of the monoid for R, which in turn equals the sum $b + c$ between the prefix and the period of R. In conclusion, Algorithm 5 runs in time at most $2^{\mathcal{O}(N^{\log_{10} 11})}$.

9. Experiments

In this section, we report on experiments we have performed in order to evaluate our transitive closure and reachability analysis algorithms.

9.1. Transitive Closure Computation

We have implemented Algorithm 5 for difference bounds and octagonal relations within the FLATA toolset [37]. We compared the performance of this algorithm with existing transitive closure computation methods for difference bounds [15] and octagonal relations [11].

Table 2 shows the results of the comparison between the older algorithms described in [15, 11] (denoted as *old*) and Algorithm 5 for difference bounds relations $d_{1,\ldots,6}$ and octagonal relations $o_{1,\ldots,6}$. The tests have been performed on both *compact* (minimum number of constraints) and *canonical* (i.e. closed, for difference bounds and tightly closed, for octagons) relations. The *speedup* column gives the ratio between the *old* and *new* execution times. The experiments were performed on a 2.53GHz machine with 4GB of memory.

Table 2: A comparison with older algorithms on difference bounds and octagons. Times are in milliseconds.

Relation		new	compact		canonical	
			old	speedup	old	speedup
d_0	$(x - x' = -1) \wedge (x = y')$	0.18	0.70	3.9	38.77	215.4
d_1	$(x - x' = -1) \wedge (x' = y')$	0.18	18.18	101.0	38.77	215.4
d_2	$(x - x' = -1) \wedge (x = y') \wedge (x - z' \leq 5) \wedge (z = z')$	1.20	26.50	22.1	33431.20	27859.3
d_3	$(x - x' = -1) \wedge (x = y') \wedge (x - z \leq 5) \wedge (z = z')$	0.60	32.70	54.5	33505.50	55841.7
d_4	$(x - x' = -1) \wedge (x = y) \wedge (x - z \leq 5) \wedge (z = z')$	0.50	702.30	1404.6	48913.80	97827.6
d_5	$(a = c) \wedge (b = a') \wedge (b = b') \wedge (c = c')$	1.80	5556.60	3087.00	$> 10^6$	∞
d_6	$(a - b' \leq -1) \wedge (a - e' \leq -2) \wedge (b - a' \leq -2)$ $\wedge (b - c' \leq -1) \wedge (c - b' \leq -2) \wedge (c - d' \leq -1)$ $\wedge (d - c' \leq -2) \wedge (d - e' \leq -1 \wedge e - a' \leq -1)$ $\wedge (e - d' \leq -2) \wedge (a' - b \leq 4) \wedge (a' - c \leq 3)$ $\wedge (b' - c \leq 4 \wedge b' - d \leq 3) \wedge (c' - d \leq 4) \wedge (c' - e \leq 3)$ $\wedge (d' - a \leq 3 \wedge d' - e \leq 4) \wedge (e' - a \leq 4) \wedge (e' - b \leq 3)$	5.6	$> 10^6$	∞	$> 10^6$	∞
o_1	$(x + x' = 1)$	0.21	0.91	4.3	0.91	4.3
o_2	$(x + y' \leq -1) \wedge (-y - x' \leq -2)$	0.29	0.85	2.9	0.84	2.9
o_3	$(x \leq x') \wedge (x + y' \leq -1) \wedge (-y - x' \leq -2)$	0.32	0.93	2.9	0.94	2.9
o_4	$(x + y \leq 5) \wedge (-x + x' \leq -2) \wedge (-y + y' \leq -3)$	0.21	3.67	17.5	13.52	64.4
o_5	$(x + y \leq 1) \wedge (-x \leq 0) \wedge (-y \leq 0)$	1.20	20050.90	16709.1	$> 10^6$	∞
o_6	$(x \geq 0) \wedge (y \geq 0) \wedge (x' \geq 0) \wedge (y' \geq 0)$ $\wedge (x + y \leq 1) \wedge (x' + y' \leq 1) \wedge (x - 1 \leq x')$ $\wedge (x' \leq x + 1) \wedge (y - 1 \leq y') \wedge (y' \leq y + 1)$	2.5	$> 10^6$	∞	$> 10^6$	∞

Table 3: Comparison with FAST (MONA plugin) on deterministic difference bounds. Times are in seconds. E_T: timeout 30 s, E_B: BDD too large, E_M: out of memory.

vars	FLATA			FAST				
	done	av.	E_T	done	av.	E_T	E_M	E_B
10	50	1.5	0	49	0.6	0	0	1
15	50	1.6	0	31	10.5	17	0	2
20	50	1.6	0	4	3.4	9	8	29
25	50	1.6	0	2	4.2	2	10	36
50	50	1.6	0	0	-	0	0	50
100	49	7.7	1	0	-	0	0	50

(a) – matrix density 3%

vars	FLATA			FAST				
	done	av.	E_T	done	av.	E_T	E_M	E_B
10	50	1.5	0	22	6.9	23	1	4
15	50	1.5	0	1	20.6	4	3	42
20	50	1.6	0	0	-	1	0	49
25	43	1.7	7	0	-	0	0	50
50	50	2.3	0	0	-	0	0	50
100	42	5.5	8	0	-	0	0	50

(b) – matrix density 10%

As shown in Table 2, the maximum observed speedup is almost 10^5 for difference bounds (d_4 in canonical form) and of the order of four for octagons. For the relations d_5 (canonical form), d_6 and o_6 the computation using older methods took longer than 10^6 msec. It is also worth noticing that the highest execution time with the new method was of 2.5 msec.

Table 3 compares FLATA with the FAST tool [7] on counter systems with one

self loop labeled with a randomly generated deterministic difference bounds relation. We generated 50 such relations for each size $N = 10, 15, 20, 25, 50, 100$. Notice that FAST usually runs out of memory for more than 25 variables, whereas FLATA can handle 100 variables in reasonable time (less than 8 seconds on average).

9.2. Reachability Analysis

We have implemented the reachability analysis based on acceleration and procedure summaries, described in Section **??**, in the FLATA verifier [37]. We use algorithms that are specific to subclasses of integer relations (e.g. difference bounds or octagonal relations) for operations such as composition, satisfiability, and transitive closure. We resort to an external SMT solver YICES [24] only for checking satisfiability of polyhedra and modulo relations.

Table 4 compares the performance of FLATA with several other reachability analysis tools based on different verification methodologies. The FAST verifier [7] is based on acceleration of loops labeled with finite monoid affine relations. We have run FAST with several available plugins for solving Presburger queries: MONA [40] (finite automata), Prestaf [23] (shared automata), and Omega [49] (quantifier elimination). Table 4 reports on PresTaf which outperformed other plugins. The ELDARICA and ARMC tools [37, 47] use predicate abstraction and interpolation-based abstraction refinement. The ASPIC tool [30] uses widening-based abstract interpretation.

The benchmarks are all in the Numerical Transition Systems format[8] (NTS). We have considered six sets of examples, extracted automatically from different sources: (a) C programs with arrays provided as examples of divergence in predicate abstraction [39], (b) verification conditions for programs with arrays, expressed in the SIL logic of [12] and translated to NTS, (c) small C programs with challenging loops, (d) NTS extracted from programs with singly-linked lists by the L2CA tool [10], (e) C programs with asynchronous procedure calls translated into NTS using the approach of [28] (the examples with extension .optim are obtained via an optimized translation method [27]), and (f) models extracted from VHDL models of circuits following the method of [53]. Table 4 also reports on the size of NTS models, some of which have multiple procedures: $\|\mathbf{x}\|$, $\|Q\|$, and $\|T\|$ denote the total number of variables, the total number of control states, and the total number of transitions of all procedures of the respective model. The platform used for experiments is Intel® Core™ 2 Duo CPU P8700, 2.53GHz with 4GB of RAM.

Next, we briefly describe some of the benchmarks we considered and then comment on the results of our experiments.

9.2.1. Benchmarks

One of the set of models we considered—denoted (f) in Table 4—is taken from [53] where an approach for verification of generic VHDL circuit designs based on translation to counter automata is presented. Traditional verification techniques for hardware systems usually assume that the state space of these systems is finite. The approach

[8] `http://richmodels.epfl.ch/ntscomp_ntslib`

Table 4: A comparison of reachability analysis tools. The letter after the model name distinguishes **C**orrect models from models with a reachable **E**rror state. Items with "-", "d", and "x" signify timeout of 300s, "don't know" answer, and an unsupported class of models, respectively.

Model	Size			Time [s]				
	$\|\mathbf{x}\|$	$\|Q\|$	$\|T\|$	FLATA	ELDARICA	FAST	ARMC	ASPIC
(a) Examples from [39]								
anubhav (C)	29	20	25	0.8	3.0	49.2	2.6	0.2
copy1 (E)	39	21	24	2.0	7.2	14.5	44.0	d
cousot (C)	29	31	34	0.6	-	35.1	4.0	0.2
loop1 (E)	34	21	24	1.7	7.1	11.6	36.1	d
loop (E)	34	21	24	1.8	5.9	17.3	36.1	d
scan (E)	32	25	29	3.3	-	9.0	-	d
string_concat1 (E)	40	43	56	5.3	-	-	-	d
string_concat (E)	34	39	52	4.9	-	-	-	d
string_copy (E)	37	30	36	4.6	-	35.6	-	d
substring1 (E)	45	49	61	0.6	9.4	-	0.8	d
substring (E)	33	33	41	2.1	3.3	-	0.4	d
(b) Verification conditions for array programs [12]								
rotation_vc.1 (C)	11	13	55	0.6	2.0	x	0.6	x
rotation_vc.2 (C)	11	20	93	1.6	2.2	x	0.7	x
rotation_vc.1 (E)	11	13	56	1.1	1.3	x	0.3	x
split_vc.1 (C)	14	32	183	3.9	3.7	x	3.8	x
split_vc.2 (C)	14	29	146	3.0	2.3	x	1.1	x
split_vc.1 (E)	14	38	276	28.5	2.3	x	1.7	x
(c) Examples from [45]								
gopan (C)	25	26	28	0.4	-	0.6	-	d
rate_limiter (C)	35	25	27	31.7	6.1	x	8.1	x
(d) Examples from L2CA [10]								
bubblesort (E)	12	674	791	14.9	9.9	-	0.9	d
insdel (E)	7	28	31	0.1	1.3	1.2	0.1	d
insertsort (E)	13	130	169	2.0	4.2	-	0.3	d
listcounter (C)	4	31	35	0.3	-	14.2	2.3	0.1
listcounter (E)	6	31	34	0.3	1.4	-	0.1	d
listreversal (C)	7	97	107	4.5	3.0	-	47.9	0.1
listreversal (E)	10	99	107	0.8	2.7	-	0.3	d
mergesort (E)	11	544	606	1.2	7.7	-	0.7	d
selectionsort (E)	15	401	459	1.5	8.1	-	0.5	d
(e) Examples from [28]								
h1 (E)	28	40	50	-	5.1	x	17.7	x
h1.optim (E)	19	38	39	0.8	2.9	x	0.7	x
h1h2 (E)	29	41	52	-	9.4	x	57.0	x
h1h2.optim (E)	20	39	41	1.1	3.3	x	3.4	x
simple (E)	28	40	50	-	6.4	x	17.2	x
simple.optim (E)	19	38	39	0.8	3.0	x	0.7	x
test0 (C)	28	41	52	-	23.0	x	58.9	x
test0.optim (C)	19	39	40	0.3	3.2	x	4.3	x
test0 (E)	27	39	48	-	5.4	x	17.4	x
test0.optim (E)	19	37	38	0.6	3.0	x	0.6	x
test1.optim (C)	24	58	62	0.9	4.7	x	23.1	x
test1.optim (E)	24	56	60	1.5	4.4	x	10.8	x
test2_1.optim (E)	22	50	55	1.6	5.2	x	6.0	x
test2_2.optim (E)	22	51	56	2.9	4.6	x	5.9	x
test2.optim (C)	37	55	78	6.4	27.2	x	93.5	x
wrpc.manual (C)	5	9	13	0.6	1.2	x	47.1	x
wrpc (E)	54	60	89	-	7.9	x	0.3	x
wrpc.optim (E)	34	49	55	-	5.1	x	1.4	x
(f) VHDL models from [53]								
counter (C)	2	6	13	0.1	1.6	0.8	0.2	0.1
register (C)	2	10	49	0.2	1.1	0.5	0.2	0.1
synlifo (C)	3	43	1006	16.6	22.1	171.8	52.8	2.6

presented in [53] aims at verification of parameterized VHDL components with infinite state space. The translation to counter automata described in [53] maps bit variables to control locations and integer variables to counters. Various safety properties are encoded as bit variables whose values are equivalent to propositional logic formulae representing the bad (unsafe) states. For instance, the SYNLIFO is a synchronous LIFO component with push and pop operations, which implements signals empty and full. The property checks if these signals are set correctly for a LIFO container of arbitrary size.

Another set of examples—denoted (b) in Table 4—are counter automata generated from programs with singly-linked lists, using the approach described in [10]. The main idea is that the set of heaps generated by a program with a finite number of local variables can be represented by a finite number of shape graphs, and the (unbounded) lengths of various list segments can be tracked by counters. The result of the translation of a program with lists is a counter automaton whose transition semantics is in bisimulation with the original program. For all singly-linked list programs, we check that there are no null pointer dereferences. For instance, the LISTREVERSAL is a textbook program that returns a list containing the same elements as the input list in the reversed order. The reversal is done in place by changing the links between the cells instead of creating a copy of the input list. Here, we also check that the lengths of the input list equals the length of the output list.

A next set of counter automata models—denoted (d) in Table 4—are obtained from the decision procedure of the array logic SIL (Singly Indexed Logic), described in [32]. The decidability of the satisfiability problem for SIL encodes the set of models of a formula as the union of sets of traces of a set of flat counter automata with difference bounds constraints, whose emptiness is known to be decidable, e.g., [19, 25]. Since FLATA is guaranteed to terminate on flat models with periodic relations on loops, we can use it as a solver for the SIL logic. We report on two SIL formulae which arise as verification conditions for loop invariants of array manipulating programs. The *array rotation* program rotates an array by one element to the left, and the *array split* program splits an array to negative and non-negative parts.

The (f) benchmarks in Table 4 were generated from C programs with asynchronous procedure calls. For instance, WRPC is a simplified asynchronous implementation of windowed RPC, in which a client makes n asynchronous procedure calls in all, of which at most $w \leq n$ are pending at any time.

The (a) models include several tricky numerical puzzles as well well as programs that manipulate C strings, e.g. programs creating copies or concatenations of strings. The translation scheme [29] generates models that detect out-of-bound errors.

9.2.2. Experimental Results

First, consider the tools FLATA and FAST which are both based on precise reachability methods that use acceleration. Table 4 shows that FLATA significantly outperforms FAST on a vast majority of benchmarks. Note that we could not make a comparison for (b) and (e) benchmarks since the FAST tool does not support transitions with non-deterministic updates like $x' \geq 2$. The ASPIC tool manifests strengths and weaknesses of abstract interpretation: correctness of models can be usually verified quickly,

however, absence of abstraction refinement often leads to "don't know" answers for models which have an error trace.

ELDARICA and ARMC are tools based on interpolation-based predicate abstraction and it turns out that they successfully verify almost same models (the sole exception being COUSOT and LISTCOUNTER models). Comparing FLATA with ELDARICA (or with ARMC), one can observe that the tools behave in a complementary way. In some cases (examples (a)), the predicate abstraction method fails due to an unbounded number of loop unrollings required by refinement. In these cases, acceleration was capable to find the needed invariant rather quickly. On the other hand (examples (e)), the acceleration approach was unsuccessful in reducing loops with linear but non-octagonal relations. In these cases, the predicate abstraction found the needed Presburger invariants for proving correctness and error traces for the erroneous examples.

References

[1] R. Alur and D. L. Dill. The theory of timed automata. In *proc. of REX Workshop*, volume 600 of *LNCS*, pages 45–73, Berlin, Heidelberg, 1991. Springer Verlag.

[2] R. Alur and P. Madhusudan. Visibly pushdown languages. In *Proc. of STOC*, pages 202–211, New York, NY, USA, 2004. ACM.

[3] R. Alur and P. Madhusudan. Adding nesting structure to words. In *Proc. of DLT*, pages 1–13, Berlin, Heidelberg, 2006. Springer-Verlag.

[4] A. Annichini, E. Asarin, and A. Bouajjani. Symbolic techniques for parametric reasoning about counter and clock systems. In *Proc. of CAV*, volume 1855 of *LNCS*, pages 419–434, Berlin, Heidelberg, 2000. Springer Verlag.

[5] R. Bagnara, P. M. Hill, and E. Zaffanella. An improved tight closure algorithm for integer octagonal constraints. In *Proc. of VMCAI*, volume 4905 of *LNCS*, pages 8–21, Berlin, Heidelberg, 2008. Springer Verlag.

[6] S. Bardin, A. Finkel, J. Leroux, and P. Schnoebelen. Flat acceleration in symbolic model checking. In *Proc. of ATVA*, volume 3707 of *LNCS*, pages 474–488, Berlin, Heidelberg, 2005. Springer Verlag.

[7] S. Bardin, J. Leroux, and G. Point. Fast extended release. In *Proc. of CAV*, volume 4144 of *LNCS*, pages 63–66, Berlin, Heidelberg, 2006. Springer Verlag.

[8] D. Beyer and M. E. Keremoglu. CPAchecker: A tool for configurable software verification. In *Proc. of CAV*, volume 6806 of *LNCS*, pages 184–190, Berlin, Heidelberg, 2011. Springer Verlag.

[9] B. Boigelot. *Symbolic Methods for Exploring Infinite State Spaces*. PhD Thesis. Université de Liège, 1999.

[10] A. Bouajjani, M. Bozga, P. Habermehl, R. Iosif, P. Moro, and T. Vojnar. Programs with lists are counter automata. In *Proc. of CAV*, volume 4144 of *LNCS*, pages 517–531, Berlin, Heidelberg, 2006. Springer Verlag.

[11] M. Bozga, C. Gîrlea, and R. Iosif. Iterating octagons. In *Proc. of TACAS*, volume 5505 of *LNCS*, pages 337–351, Berlin, Heidelberg, 2009. Springer Verlag.

[12] M. Bozga, P. Habermehl, R. Iosif, F. Konečný, and T. Vojnar. Automatic verification of integer array programs. In *Proc. of CAV*, volume 5643 of *LNCS*, pages 157–172, Berlin, Heidelberg, 2009. Springer Verlag.

[13] M. Bozga and R. Iosif. On flat programs with lists. In *Proc. of VMCAI*, volume 4349 of *LNCS*, pages 122–136, Berlin, Heidelberg, 2007. Springer Verlag.

[14] M. Bozga, R. Iosif, and F. Konečný. Fast acceleration of ultimately periodic relations. In *Proc. of CAV*, volume 6174 of *LNCS*, pages 227–242, Berlin, Heidelberg, 2010. Springer Verlag.

[15] M. Bozga, R. Iosif, and Y. Lakhnech. Flat parametric counter automata. *Fundamenta Informaticae*, 91(2):275–303, 2009.

[16] M. Bozga, R. Iosif, and S. Perarnau. Quantitative separation logic and programs with lists. In *Proc. of IJCAR*, volume 5195 of *LNCS*, pages 34–49, Berlin, Heidelberg, 2008. Springer Verlag.

[17] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement for symbolic model checking. *Journal of the ACM*, 50(5):752–794, 2003.

[18] E.M. Clarke and E.A. Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In *Proc. of Logic of Programs*, volume 131 of *LNCS*, pages 52–71, Berlin, Heidelberg, 1982. Springer Verlag.

[19] H. Comon and Y. Jurski. Multiple counters automata, safety analysis and presburger arithmetic. In *Proc. of CAV*, volume 1427 of *LNCS*, pages 268–279, Berlin, Heidelberg, 1998. Springer Verlag.

[20] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.

[21] P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proc. of POPL*, pages 238–252, New York, NY, USA, 1977. ACM.

[22] P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. In *Conference Record of the Fifth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 84–97, Tucson, Arizona, 1978. ACM Press, New York, NY.

[23] J.M. Couvreur. PresTAF. `http://altarica.labri.fr/forge/projects/3/wiki/PresTAF`.

[24] B. Dutertre and L. de Moura. The YICES SMT Solver. `http://yices.csl.sri.com/`.

[25] A. Finkel and J. Leroux. How to compose presburger-accelerations: Applications to broadcast protocols. In *Proc. of FST TCS*, volume 2556 of *LNCS*, pages 145–156, Berlin, Heidelberg, 2002. Springer Verlag.

[26] A. Finkel, E. Lozes, and A. Sangnier. Towards model-checking programs with lists. In *Proc. of ILC*, volume 5489 of *LNCS*, pages 56–86, Berlin, Heidelberg, 2007. Springer Verlag.

[27] P. Ganty. Personal communication, 2012.

[28] P. Ganty and R. Majumdar. Algorithmic verification of asynchronous programs. *ACM Trans. Program. Lang. Syst.*, 34(1):6:1–6:48, 2012.

[29] F. Garnier and R. Iosif. Personal communication, 2012.

[30] L. Gonnord. ASPIC. `http://laure.gonnord.org/pro/aspic/aspic.html`.

[31] S. Graf and H. Saïdi. Construction of abstract state graphs with PVS. In *Proc. of CAV*, volume 1254 of *LNCS*, pages 72–83, Berlin, Heidelberg, 1997. Springer Verlag.

[32] P. Habermehl, Radu I., and T. Vojnar. A logic of singly indexed arrays. In *Proc. of LPAR*, volume 5330 of *LNCS*, pages 558–573, Berlin, Heidelberg, 2008. Springer Verlag.

[33] P. Habermehl, R. Iosif, A. Rogalewicz, and T. Vojnar. Proving termination of tree manipulating programs. In *Proc. of ATVA*, volume 4762 of *LNCS*, pages 145–161, Berlin, Heidelberg, 2007. Springer Verlag.

[34] P. Habermehl, R. Iosif, and T. Vojnar. What else is decidable about integer arrays? In *Proc. of FoSSaCS*, volume 4962 of *LNCS*, pages 474–489, Berlin, Heidelberg, 2008. Springer Verlag.

[35] M. Heizmann, J. Hoenicke, and A. Podelski. Nested interpolants. In *Proc. of POPL*, pages 471–482, New York, NY, USA, 2010. ACM.

[36] T. A. Henzinger, R. Jhala, R. Majumdar, and G. Sutre. Software verification with BLAST. In *Proc. of SPIN*, volume 2648 of *LNCS*, pages 235–239, Berlin, Heidelberg, 2003. Springer Verlag.

[37] H. Hojjat, R. Iosif, F. Garnier, F. Konečný, V. Kuncak, and P. Rümmer. A verification toolkit for numerical transition systems. In *Proc. of FM*, 2012. To appear.

[38] O. H. Ibarra. Reversal-bounded multicounter machines and their decision problems. *Journal of the ACM*, 25(1):116–133, January 1978.

[39] R. Jhala and K. L. McMillan. A practical and complete approach to predicate refinement. In *Proc. of TACAS*, volume 3920 of *LNCS*, pages 459–473, Berlin, Heidelberg, 2006. Springer Verlag.

[40] Nils Klarlund and Anders Møller. MONA. `http://www.brics.dk/mona/`.

[41] J. Leroux and G. Sutre. Flat counter automata almost everywhere! In *Proc. of ATVA*, volume 3707 of *LNCS*, pages 489–503, Berlin, Heidelberg, 2005. Springer Verlag.

[42] A. Mandel and I. Simon. On finite semigroups of matrices. *Theoretical Computer Science*, 5(2):101–111, 1977.

[43] A. Miné. The octagon abstract domain. *Higher-Order and Symbolic Computation*, 19(1):31–100, 2006.

[44] M. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, 1967.

[45] D. Monniaux. Personal Communication, 2012.

[46] A. Podelski and A. Rybalchenko. Transition predicate abstraction and fair termination. In *Proc. of POPL*, pages 132–144, New York, NY, USA, 2005. ACM.

[47] A. Podelski and A. Rybalchenko. ARMC: The logical choice for software model checking with abstraction refinement. In *Proc. of PADL*, volume 4354 of *LNCS*, pages 245–259. Springer Verlag, Berlin, Heidelberg, 2007.

[48] M. Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. *Comptes rendus du I Congrés des Pays Slaves*, page 92101, 1929.

[49] W. Pugh, E. Rosser, W. Kelly, D. Wonnacott, and T. Shpeisman. Omega. `http://www.cs.umd.edu/projects/omega/`.

[50] J.-P. Queille and J. Sifakis. Specification and verification of concurrent systems in cesar. In *Proc. of the 5th Colloquium on International Symposium on Programming*, volume 137 of *LNCS*, pages 337–351, Berlin, Heidelberg, 1982. Springer Verlag.

[51] C. Reutenauer. *The mathematics of Petri nets*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1990.

[52] B. De Schutter. On the ultimate behavior of the sequence of consecutive powers of a matrix in the max-plus algebra. *Linear Algebra and its Applications*, 307:103–117, 2000.

[53] A. Smrcka and T. Vojnar. Verifying parametrised hardware designs via counter automata. In *Proc. of HVC*, volume 4899 of *LNCS*, pages 51–68, Berlin, Heidelberg, 2007. Springer Verlag.