Decision Problems in a Logic for Reasoning about Reconfigurable Distributed Systems

Marius Bozga, Lucas Bueri, Radu Iosif

Decision Problems in a Logic for Reasoning about Reconfigurable Distributed Systems

Marius Bozga[1], Lucas Bueri[1], and Radu Iosif[1]

Univ. Grenoble Alpes, CNRS, Grenoble INP, VERIMAG, 38000, France

Abstract. We consider a logic used to describe sets of configurations of distributed systems, whose network topologies can be changed at runtime, by reconfiguration programs. The logic uses inductive definitions to describe networks with an unbounded number of components and interactions, written using a multiplicative conjunction, reminiscent of Bunched Implications [38] and Separation Logic [40]. We study the complexity of the satisfiability and entailment problems for the configuration logic under consideration. Additionally, we consider robustness properties, such as tightness (are all interactions entirely connected to components?) and degree boundedness (is every component involved in a bounded number of interactions?), the latter being an ingredient for decidability of entailments.

1 Introduction

Distributed systems are increasingly used as critical parts of the infrastructure of our digital society, as in e.g., datacenters, e-banking and social networking. In order to address maintenance (e.g., replacement of faulty and obsolete network nodes by new ones) and data traffic issues (e.g., managing the traffic inside a datacenter [36]), the distributed systems community has recently put massive effort in designing algorithms for *reconfigurable systems*, whose network topologies change at runtime [25]. However, dynamic reconfiguration is an important souce of bugs that may result in e.g., denial of services or even data corruption[1].

This paper contributes to a logical framework that addresses the timely problems of formal *modeling* and *verification* of reconfigurable distributed systems. The basic building blocks of this framework are (i) a Hoare-style program proof calculus [1] used to write formal proofs of correctness of reconfiguration programs, and (ii) an invariant synthesis method [6] that proves the safety (i.e., absence of reachable error configurations) of the configurations defined by the assertions that annotate a reconfiguration program. These methods are combined to prove that an initially correct distributed system cannot reach an error state, following the execution of a given reconfiguration sequence.

The assertions of the proof calculus are written in a logic that defines infinite sets of configurations, consisting of *components* (i.e., processes running on different nodes of the network) connected by *interactions* (i.e., multi-party channels alongside which messages between components are transfered). Systems that share the same architectural style (e.g., pipeline, ring, star, tree, etc.) and differ by the number of components

[1] `https://status.cloud.google.com/incident/appengine/19007`

and interactions are described using inductively defined predicates. Such configurations can be modified either by (a) adding or removing components and interactions (reconfiguration), or (b) changing the local states of components, by firing interactions.

The assertion logic views components and interactions as *resources*, that can be created or deleted, in the spirit of resource logics *à la* Bunched Implications [38], or Separation Logic [40]. The main advantage of using resource logics is their support for *local reasoning* [12]: reconfiguration actions are specified by pre- and postconditions mentioning only the resources involved, while framing out the rest of the configuration.

The price to pay for this expressive power is the difficulty of automating the reasoning in these logics. This paper makes several contributions in the direction of proof automation, by studying the complexity of the *satisfiability* and *entailment* problems, for the configuration logic under consideration. Additionally, we study the complexity of two robustness properties [28], namely *tightness* (are all interactions entirely connected to components?) and *degree boundedness* (is every component involved in a bounded number of interactions?). In particular, the latter problem is used as a prerequisite for defining a fragment with a decidable entailment problem.

1.1 Motivating Example

The logic studied in this paper is motivated by the need for an assertion language that supports reasoning about dynamic reconfigurations in a distributed system. For instance, consider a distributed system consisting of a finite (but unknown) number of *components* (processes) placed in a ring, executing the same finite-state program and communicating via *interactions* that connect the *out* port of a component to the *in* port of its right neighbour, in a round-robin fashion, as in Fig. 1 (a). The behavior of a component is a machine with two states, T and H, denoting whether the component has a token (T) or not (H). A component c_i without a token may receive one, by executing a transition $\mathsf{H} \xrightarrow{in} \mathsf{T}$, simultaneously with its left neighbour c_j, that executes the transition transition $\mathsf{T} \xrightarrow{out} \mathsf{H}$. Then, we say that the interaction (c_j, out, c_i, in) has fired, moving a token one position to the right in the ring. Note that there can be more than one token, moving independently in the system, as long as no token overtakes another token.

The token ring system is formally specified by the following inductive rules:

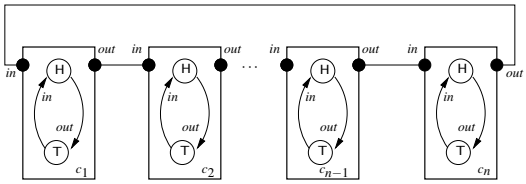
$$\mathsf{ring}_{h,t}(x) \leftarrow \exists y \exists z \;.\; [x]@q * \langle x.out, z.in \rangle * \mathsf{chain}_{h',t'}(z,y) * \langle y.out, x.in \rangle$$

$$\mathsf{chain}_{h,t}(x,y) \leftarrow \exists z.\; [x]@q * \langle x.out, z.in \rangle * \mathsf{chain}_{h',t'}(z,y)$$

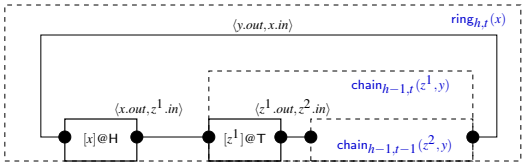
$$\mathsf{chain}_{0,1}(x,x) \leftarrow [x]@\mathsf{T} \qquad \mathsf{chain}_{1,0}(x,x) \leftarrow [x]@\mathsf{H} \qquad \mathsf{chain}_{0,0}(x,x) \leftarrow [x]$$

$$\text{where } h' \stackrel{\text{def}}{=} \begin{cases} \max(h-1,0) & \text{, if } q = \mathsf{H} \\ h & \text{, if } q = \mathsf{T} \end{cases} \text{ and } t' \stackrel{\text{def}}{=} \begin{cases} \max(t-1,0) & \text{, if } q = \mathsf{T} \\ t & \text{, if } q = \mathsf{H} \end{cases}$$

The predicate $\mathsf{ring}_{h,t}(x)$ describes a ring with at least two components, such that at least h (resp. t) components are in state H (resp. T). The ring consists of a component x in state q, described by the formula $[x]@q$, an interaction from the *out* port of x to the *in* port of another component z, described as $\langle x.out, z.in \rangle$, a separate chain of components stretching from z to y ($\mathsf{chain}_{h',t'}(z,y)$), and an interaction connecting the *out* port of component y to the *in* port of component x ($\langle y.out, x.in \rangle$). Inductively, a chain consists of



(a)



(b)

$\{\mathsf{ring}_{h,t}(y)\}$ /*assuming $h \geq 2,\ t \geq 1$*/
$\{\boxed{[y]@\mathsf{H} * \langle y.out, z.in \rangle * \mathsf{chain}_{h-1,t}(z,x)} * \langle x.out, y.in \rangle\}$
`disconnect(x.`out`, y.`in`);`
$\{\boxed{[y]@\mathsf{H}} * \langle y.out, z.in \rangle * \boxed{\mathsf{chain}_{h-t,t}(z,x)}\}$
`disconnect(y.`out`, z.`in`);`
$\{[y]@\mathsf{H} * \boxed{\mathsf{chain}_{h-1,t}(z,x)}\}$
`delete(y);`
$\{\boxed{\mathsf{chain}_{h-1,t}(z,x)}\}$
`connect(x.`out`, z.`in`)`
$\{\mathsf{chain}_{h-1,t}(z,x) * \langle x.out, z.in \rangle\}$
$\{\mathsf{ring}_{h-1,t}(z)\}$

(c)

Fig. 1: Inductive Specification and Reconfiguration of a Token Ring

a component $[x]@q$, an interaction $\langle x.out, z.in \rangle$ and a separate $\mathsf{chain}_{h',t'}(z,y)$. Fig. 1 (b) depicts the unfolding of the inductive definition of the token ring, with the existentially quantified variables z from the above rules α-renamed to $z^1, z^2, \ldots$ to avoid confusion.

A *reconfiguration program* takes as input a mapping of program variables to components and executes a sequence of *basic operations* i.e., component/interaction creation/deletion, involving the components and interactions denoted by these variables. For instance, the reconfiguration program in Fig. 1 (c) takes as input three adjacent components, mapped to the variables x, y and z, respectively, removes the component y together with its left and right interactions and reconnects x directly with z. Programming reconfigurations is error-prone, because the interleaving between reconfiguration actions and interactions in a distributed system may lead to bugs that are hard to trace. For instance, if a reconfiguration program removes the last component in state T (resp. H) from the system, no token transfer interaction may fire and the system deadlocks.

We prove absence of such errors using a Hoare-style proof system [1], based on the above logic as assertion language. For instance, the proof from Fig. 1 (c) shows that the reconfiguration sequence applied to a component y in state H (i.e., $[y]@\mathsf{H}$) in a ring with at least $h \geq 2$ components in state H and at least $t \geq 1$ components in state T leads to a ring with at least $h-1$ components in state H and at least t in state T; note that the states of the components may change during the execution of the reconfiguration program, as tokens are moved by interactions.

For reasons of proof scalability, a basic operation is specified only with regard to the components and interactions required to avoid faulting. For instance $\{[x]@q\}\ \mathsf{delete}(\mathsf{x})\ \{\mathsf{emp}\}$ (resp. $\{\langle x.out, y.in \rangle\}\ \mathsf{disconnect}(\mathtt{x}.out, \mathtt{y}.in)\ \{\mathsf{emp}\}$) means that delete (resp. disconnect) requires a component (resp. interaction) and returns an empty configuration, whereas $\{\mathsf{emp}\}\ \mathsf{connect}(\mathtt{x}.out, \mathtt{y}.in)\ \{\langle x.out, y.in \rangle\}$ means that connect requires nothing and creates an interaction between the given ports of the given components. These local specifications are plugged into a context described by a frame formula F, using the *frame rule* $\{\phi\}\ \mathsf{P}\ \{\psi\} \Rightarrow \{\phi * \boxed{F}\}\ \mathsf{P}\ \{\psi * F\}$; for readability, the frame formulæ (from the preconditions of the conclusion of the frame rule applications) are enclosed in boxes.

The proof also uses the *consequence rule* $\{\phi\}\ \mathsf{P}\ \{\psi\} \Rightarrow \{\phi'\}\ \mathsf{P}\ \{\psi'\}$ that applies if ϕ' is stronger than ϕ and ψ' is weaker than ψ. The side conditions of the consequence rule require checking the validity of the entailments $\mathsf{ring}_{h,t}(y) \models \exists x \exists z\ .\ \langle x.out, y.in\rangle * [y]@\mathsf{H} * \langle y.out, z.in\rangle * \mathsf{chain}_{h-1,t}(z,x)$ and $\mathsf{chain}_{h-1,t}(z,x) * \langle x.out, z.in\rangle \models \mathsf{ring}_{h-1,t}(z)$, for all $h \geq 2$ and $t \geq 1$. These side conditions can be automatically discharged using the results on the decidability of entailments given in this paper. Additionally, checking the satisfiability of a precondition is used to detect trivially valid Hoare triples.

1.2 Related Work

Formal modeling coordinating architectures of component-based systems has received lots of attention, with the development of architecture description languages (ADL), such as BIP [4] or REO [2]. Many such ADLs have extensions that describe programmed reconfiguration, e.g., [21,31], classified according to the underlying formalism used to define their operational semantics: *process algebras* [13,34], *graph rewriting* [42,45,33], *chemical reactions* [44] (see the surveys [7,11]). Unfortunately, only few ADLs support formal verification, mainly in the flavour of runtime verification [10,17,32,22] or finite-state model checking [14].

Parameterized verification of unbounded networks of distributed processes uses mostly hard-coded coordinating architectures (see [5] for a survey). A first attempt at specifying architectures by logic is the *interaction logic* of Konnov et al. [30], a combination of Presburger arithmetic with monadic uninterpreted function symbols, that can describe cliques, stars and rings. More structured architectures (pipelines and trees) can be described using a second-order extension [35]. However, these interaction logics are undecidable and lack support for automated reasoning.

Specifying parameterized component-based systems by inductive definitions is not new. *Network grammars* [41,33,27] use context-free grammar rules to describe systems with linear (pipeline, token-ring) architectures obtained by composition of an unbounded number of processes. In contrast, we use predicates of unrestricted arities to describe architectural styles that are, in general, more complex than trees. Moreover, we write inductive definitions using a resource logic, suitable also for writing Hoare logic proofs of reconfiguration programs, based on local reasoning [12].

Local reasoning about concurrent programs has been traditionally the focus of Concurrent Separation Logic (CSL), based on a parallel composition rule [37], initially with a non-interfering (race-free) semantics [8] and later combining ideas of assume-and rely-guarantee [39,29] with local reasoning [24,43] and abstract notions of framing [16,15,23]. However, the body of work on CSL deals almost entirely with shared-memory multithreading programs, instead of distributed systems, which is the aim of our work. In contrast, we develop a resource logic in which the processes do not just share and own resources, but become mutable resources themselves.

The techniques developed in this paper are inspired by existing techniques for similar problems in the context of Separation Logic (SL) [40]. For instance, we use an abstract domain similar to the one defined by Brotherston et al. [9] for checking satisfiability of symbolic heaps in SL and reduce a fragment of the entailment problem in our logic to SL entailment [20]. In particular, the use of existing automated reasoning techniques for SL has pointed out several differences between the expressiveness of our

logic and that of SL. First, the configuration logic describes hypergraph structures, in which edges are ℓ-tuples for $\ell \geq 2$, instead of directed graphs as in SL, where ℓ is a parameter of the problem: considering ℓ to be a constant strictly decreases the complexity of the problem. Second, the degree (number of hyperedges containing a given vertex) is unbounded, unlike in SL, where the degree of heaps is constant. Therefore, we dedicate an entire section (§5) to the problem of deciding the existence of a bound (and computing a cut-off) on the degree of the models of a formula, used as a prerequisite for the encoding of the entailment problems from the configuration logic as SL entailments.

2 Definitions

We denote by $\mathbb{N}$ the set of positive integers. For a set A, we define $A^1 \stackrel{\text{def}}{=} A$, $A^{i+1} \stackrel{\text{def}}{=} A^i \times A$, for all $i \geq 0$, and $A^+ = \bigcup_{i \geq 1} A^i$, where $\times$ denotes the Cartesian product. We denote by $\text{pow}(A)$ the powerset of A and by $\text{mpow}(A)$ the power-multiset (set of multisets) of A. The cardinality of a finite set A is denoted as $\|A\|$. By writing $A \subseteq_{fin} B$ we mean that A is a finite subset of B. Given integers i and j, we write $[i, j]$ for the set $\{i, i+1, \ldots, j\}$, assumed to be empty if $i > j$. For a tuple $\mathbf{t} = \langle t_1, \ldots, t_n \rangle$, we define $|\mathbf{t}| \stackrel{\text{def}}{=} n$, $\langle \mathbf{t} \rangle_i \stackrel{\text{def}}{=} t_i$ and $\langle \mathbf{t} \rangle_{[i,j]} \stackrel{\text{def}}{=} \langle t_i, \ldots, t_j \rangle$. By writing $x = poly(y)$, for given $x, y \in \mathbb{N}$, we mean that there exists a polynomial function $f : \mathbb{N} \to \mathbb{N}$, such that $x \leq f(y)$.

2.1 Configurations

We model distributed systems as hypergraphs, whose vertices are *components* (i.e., the nodes of the network) and hyperedges are *interactions* (i.e., describing the way the components communicate with each other). The components are taken from a countably infinite set $\mathbb{C}$, called the *universe*. We consider that each component executes its own copy of the same *behavior*, represented as a finite-state machine $\mathbb{B} = (\mathcal{P}, Q, \rightarrow)$, where $\mathcal{P}$ is a finite set of *ports*, Q is a finite set of *states* and $\rightarrow \subseteq Q \times \mathcal{P} \times Q$ is a transition relation. Intuitively, each transition $q \xrightarrow{p} q'$ of the behavior is triggerred by a visible event, represented by the port p. For instance, the behavior of the components of the token ring system from Fig. 1 (a) is $\mathbb{B} = (\{in, out\}, \{\mathsf{H}, \mathsf{T}\}, \{\mathsf{H} \xrightarrow{in} \mathsf{T}, \mathsf{T} \xrightarrow{out} \mathsf{H}\})$. The universe $\mathbb{C}$ and the behavior $\mathbb{B} = (\mathcal{P}, Q, \rightarrow)$ are considered fixed in the rest of this paper.

We introduce a logic for describing infinite sets of *configurations* of distributed systems with unboundedly many components and interactions. A configuration is a snapshot of the system, describing the topology of the network (i.e., the set of present components and interactions) together with the local state of each component:

Definition 1. *A* configuration *is a tuple* $\gamma = (C, I, \rho)$*, where:*

- $C \subseteq_{fin} \mathbb{C}$ *is a finite set of* components*, that are present in the configuration,*
- $I \subseteq_{fin} (\mathbb{C} \times \mathcal{P})^+$ *is a finite set of* interactions*, where each interaction is a sequence* $(c_1, p_1, \ldots, c_n, p_n) \in (\mathbb{C} \times \mathcal{P})^n$ *that binds together the ports* $p_1, \ldots, p_n$ *of the pairwise distinct components* $c_1, \ldots, c_n$*, respectively.*
- $\rho : \mathbb{C} \to Q$ *is a* state map *associating each (possibly absent) component, a state of the behavior* $\mathbb{B}$*, such that the set* $\{c \in \mathbb{C} \mid \rho(c) = q\}$ *is infinite, for each* $q \in Q$*.*

The last condition requires that there is an infinite pool of components in each state $q \in Q$; since $\mathbb{C}$ is infinite and Q is finite, this condition is feasible. For example, the configurations of the token ring from Fig. 1 (a) are $(\{c_1,\ldots,c_n\}, \{(c_i, out, c_{(i \bmod n)+1}, in) \mid i \in [1,n]\}, \rho)$, where $\rho : \mathbb{C} \to \{\mathsf{H},\mathsf{T}\}$ is a state map. The ring topology is described by the set of components $\{c_1,\ldots,c_n\}$ and interactions $\{(c_i, out, c_{(i \bmod n)+1}, in) \mid i \in [1,n]\}$.

Intuitively, an interaction $(c_1, p_1, \ldots, c_n, p_n)$ synchronizes transitions labeled by the ports $p_1,\ldots,p_n$ from the behaviors (i.e., replicas of the state machine $\mathbb{B}$) of $c_1,\ldots,c_n$, respectively. The interactions are classified according to their sequence of ports, called the *interaction type* and let $\mathsf{Inter} \stackrel{\text{def}}{=} \mathcal{P}^+$ be the set of interaction types; an interaction type models, for instance, the passing of a certain kind of message (e.g., request, acknowledgement, etc.). From an operational point of view, two interactions that differ by a permutation of indices e.g., $(c_1, p_1, \ldots, c_n, p_n)$ and $(c_{i_1}, p_{i_1}, \ldots, c_{i_n}, p_{i_n})$ such that $\{i_1,\ldots,i_n\} = [1,n]$, are equivalent, since the set of transitions is the same; nevertheless, we chose to distinguish them in the following, exclusively for reasons of simplicity.

Note that Def. 1 allows configurations with interactions that involve absent components (i.e., not from the set $\mathcal{C}$ of present components in the given configuration). The following definition distinguishes such configurations:

Definition 2. *Let $\gamma = (\mathcal{C}, I, \rho)$ be a configuration. An interaction $(c_1, p_1, \ldots, c_n, p_n)$ is* loose *in γ if and only if $c_i \notin \mathcal{C}$, for some $i \in [1,n]$. If I contains at least one interaction that is loose in γ, we say that γ is* loose. *An interaction (resp. configuration) that is not loose is said to be* tight.

For instance, every configuration of the system from Fig. 1 (a) is tight and becomes loose if a component is deleted. Moreover, the reconfiguration program from Fig. 1 (c) manipulates tight configurations only. In particular, loose configurations are useful for the definition of a composition operation, as the union of disjoint sets of components and interactions:

Definition 3. *The composition of two configurations $\gamma_i = (\mathcal{C}_i, I_i, \rho)$, for $i = 1,2$, such that $\mathcal{C}_1 \cap \mathcal{C}_2 = \emptyset$ and $I_1 \cap I_2 = \emptyset$, is defined as $\gamma_1 \bullet \gamma_2 \stackrel{\text{def}}{=} (\mathcal{C}_1 \cup \mathcal{C}_2, I_1 \cup I_2, \rho)$. The composition $\gamma_1 \bullet \gamma_2$ is undefined if $\mathcal{C}_1 \cap \mathcal{C}_2 \neq \emptyset$ or $I_1 \cap I_2 \neq \emptyset$.*

Note that a tight configuration may be the result of composing two loose configurations, whereas the composition of tight configurations is always tight. The example below shows that, in most cases, a non-trivial decomposition of a tight configuration necessarily involves loose configurations:

Example 1. Let $\gamma_i = (\mathcal{C}_i, I_i, \rho)$ be loose configurations, where $\mathcal{C}_i = \{c_i\}$, $I_i = \{(c_i, out, c_{3-i}, in)\}$, for all $i = 1,2$. Then $\gamma \stackrel{\text{def}}{=} \gamma_1 \bullet \gamma_2$ is the tight configuration $\gamma = (\{c_1,c_2\}, \{(c_1, out, c_2, in), (c_2, out, c_1, in)\}, \rho)$. The only way of decomposing γ into two tight subconfigurations γ'_1 and γ'_2 is taking $\gamma'_1 \stackrel{\text{def}}{=} \gamma$ and $\gamma'_2 \stackrel{\text{def}}{=} (\emptyset, \emptyset, \rho)$, or viceversa. ■

In analogy with graphs, the *degree* of a configuration is the maximum number of interactions from the configuration that involve a (possibly absent) component:

Definition 4. *The* degree *of a configuration $\gamma = (\mathcal{C}, I, \rho)$ is defined as $\delta(\gamma) \stackrel{\text{def}}{=} \max_{c \in \mathbb{C}} \delta_c(\gamma)$, where $\delta_c(\gamma) \stackrel{\text{def}}{=} \|\{(c_1, p_1, \ldots, c_n, p_n) \in I \mid c = c_i,\ i \in [1,n]\}\|$.*

For instance, the configuration of the system from Fig. 1 (a) has degree two.

2.2 Configuration Logic

Let $\mathbb{V}$ and $\mathbb{A}$ be countably infinite sets of *variables* and *predicates*, respectively. For each predicate $\mathsf{A} \in \mathbb{A}$, we denote its arity by $\#\mathsf{A}$. The formulæ of the *Configuration Logic* (CL) are described inductively by the following syntax:

$$\phi := \mathsf{emp} \mid [x] \mid \langle x_1.p_1, \ldots, x_n.p_n \rangle \mid x@q \mid x = y \mid x \neq y \mid \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \mid \phi * \phi \mid \exists x \,.\, \phi$$

where $x, y, x_1, \ldots \in \mathbb{V}$, $q \in Q$ and $\mathsf{A} \in \mathbb{A}$. A formula $[x]$, $\langle x_1.p_1, \ldots, x_n.p_n \rangle$, $x@q$ and $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$ is called a *component*, *interaction*, *state* and *predicate* atom, respectively. Sometimes, we use the shorthand $[x]@q \stackrel{\mathsf{def}}{=} [x] * x@q$. Intuitively, the formula $[x]@q * [y]@q' * \langle x.out, y.in \rangle * \langle x.in, y.out \rangle$ describes a configuration consisting of two distinct components, denoted by the values of x and y, in states q and q', respectively, and two interactions binding the *out* port of one to the *in* port of the other component. For instance, $\gamma = \gamma_1 \bullet \gamma_2$ from Example 1 is such a configuration.

A formula is said to be *pure* if and only if it consists of state atoms, equalities and disequalities. A formula with no occurrences of predicate atoms (resp. existential quantifiers) is called *predicate-free* (resp. *quantifier-free*). A variable is *free* if it does not occur within the scope of an existential quantifier and let $\mathrm{fv}(\phi)$ be the set of free variables of ϕ. A *sentence* is a formula with no free variables. A *substitution* $\phi[x_1/y_1 \ldots x_n/y_n]$ replaces simultaneously every free occurrence of x_i by y_i in ϕ, for all $i \in [1, n]$. Before defining the semantics of CL formulæ, we introduce the set of inductive definitions that assigns meaning to predicates:

Definition 5. *A* set of inductive definitions (SID) Δ *consists of* rules $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \phi$*, where* $x_1, \ldots, x_{\#\mathsf{A}}$ *are pairwise distinct variables, called* parameters*, such that* $\mathrm{fv}(\phi) \subseteq \{x_1, \ldots, x_{\#\mathsf{A}}\}$*. The rule* $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \phi$ defines A *and we denote by* $\mathrm{def}_\Delta(\mathsf{A})$ *the set of rules from* Δ *that define* A*.*

Note that having distinct parameters in a rule is without loss of generality, as e.g., a rule $\mathsf{A}(x_1, x_1) \leftarrow \phi$ can be equivalently written as $\mathsf{A}(x_1, x_2) \leftarrow x_1 = x_2 * \phi$. As a convention, we shall always use the names $x_1, \ldots, x_{\#\mathsf{A}}$ for the parameters of a rule that defines A.

The semantics of CL formulæ is defined by a satisfaction relation $\gamma \models^{\mathsf{v}}_{\Delta} \phi$ between configurations and formulæ. This relation is parameterized by a *store* $\mathsf{v} : \mathbb{V} \to \mathbb{C}$ mapping the free variables of a formula into components from the universe (possibly absent from γ) and an SID Δ. We write $\mathsf{v}[x \leftarrow c]$ for the store that maps x into c and agrees with v on all variables other than x. The definition of the satisfaction relation is by induction on the structure of formulæ, where $\gamma = (\mathcal{C}, I, \rho)$ is a configuration (Def. 1):

$$\begin{array}{lll}
\gamma \models^{\mathsf{v}}_{\Delta} \mathsf{emp} & \iff & \mathcal{C} = \emptyset \text{ and } I = \emptyset \\
\gamma \models^{\mathsf{v}}_{\Delta} [x] & \iff & \mathcal{C} = \{\mathsf{v}(x)\} \text{ and } I = \emptyset \\
\gamma \models^{\mathsf{v}}_{\Delta} \langle x_1.p_1, \ldots, x_n.p_n \rangle & \iff & \mathcal{C} = \emptyset \text{ and } I = \{(\mathsf{v}(x_1), p_1, \ldots, \mathsf{v}(x_n), p_n)\} \\
\gamma \models^{\mathsf{v}}_{\Delta} x@q & \iff & \gamma \models^{\mathsf{v}}_{\Delta} \mathsf{emp} \text{ and } \rho(\mathsf{v}(x)) = q \\
\gamma \models^{\mathsf{v}}_{\Delta} x \sim y & \iff & \gamma \models^{\mathsf{v}}_{\Delta} \mathsf{emp} \text{ and } \mathsf{v}(x) \sim \mathsf{v}(y), \text{ for all } \sim \in \{=, \neq\} \\
\gamma \models^{\mathsf{v}}_{\Delta} \mathsf{A}(y_1, \ldots, y_{\#\mathsf{A}}) & \iff & \gamma \models^{\mathsf{v}}_{\Delta} \phi[x_1/y_1, \ldots, x_{\#\mathsf{A}}/y_{\#\mathsf{A}}], \text{ for some rule} \\
 & & \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \phi \text{ from } \Delta \\
\gamma \models^{\mathsf{v}}_{\Delta} \phi_1 * \phi_2 & \iff & \text{exist } \gamma_1, \gamma_2, \text{ such that } \gamma = \gamma_1 \bullet \gamma_2 \text{ and } \gamma_i \models^{\mathsf{v}}_{\Delta} \phi_i, \text{ for } i = 1, 2 \\
\gamma \models^{\mathsf{v}}_{\Delta} \exists x \,.\, \phi & \iff & \gamma \models^{\mathsf{v}[x \leftarrow c]}_{\Delta} \phi, \text{ for some } c \in \mathbb{C}
\end{array}$$

If ϕ is a sentence, the satisfaction relation $\gamma \models^{\nu}_{\Delta} \phi$ does not depend on the store, written $\gamma \models_{\Delta} \phi$, in which case we say that γ is a *model* of ϕ. If ϕ is a predicate-free formula, the satisfaction relation does not depend on the SID, written $\gamma \models^{\nu} \phi$. A formula ϕ is *satisfiable* if and only if the sentence $\exists x_1 \ldots \exists x_n \,.\, \phi$ has a model, where $\mathrm{fv}(\phi) = \{x_1, \ldots, x_n\}$. A formula ϕ *entails* a formula ψ, written $\phi \models_{\Delta} \psi$ if and only if, for any configuration γ and store ν, we have $\gamma \models^{\nu}_{\Delta} \phi$ only if $\gamma \models^{\nu}_{\Delta} \psi$.

2.3 Separation Logic

Separation Logic (SL) [40] will be used in the following to prove several technical results concerning the decidability and complexity of certain decision problems for CL. For self-containment reasons, we define SL below. The syntax of SL formulæ is described by the following grammar:

$$\phi := \mathsf{emp} \mid x_0 \mapsto (x_1, \ldots, x_{\mathfrak{K}}) \mid x = y \mid x \neq y \mid \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \mid \phi * \phi \mid \exists x \,.\, \phi$$

where $x, y, x_0, x_1, \ldots \in \mathbb{V}$, $\mathsf{A} \in \mathbb{A}$ and $\mathfrak{K} \geq 1$ is an integer constant. Formulæ of SL are interpreted over finite partial functions $\mathsf{h} : \mathbb{C} \rightharpoonup_{fin} \mathbb{C}^{\mathfrak{K}}$, called *heaps*[2], by a satisfaction relation $\mathsf{h} \Vdash^{\nu} \phi$, defined inductively as follows:

$$\begin{array}{lcl}
\mathsf{h} \Vdash^{\nu}_{\Delta} \mathsf{emp} & \iff & \mathsf{h} = \emptyset \\
\mathsf{h} \Vdash^{\nu}_{\Delta} x_0 \mapsto (x_1, \ldots, x_{\mathfrak{K}}) & \iff & \mathrm{dom}(\mathsf{h}) = \{\nu(x_0)\} \text{ and } \mathsf{h}(\nu(x_0)) = \langle \nu(x_1), \ldots, \nu(x_{\mathfrak{K}}) \rangle \\
\mathsf{h} \Vdash^{\nu} \phi_1 * \phi_2 & \iff & \text{there exist } \mathsf{h}_1, \mathsf{h}_2 \text{ such that } \mathrm{dom}(\mathsf{h}_1) \cap \mathrm{dom}(\mathsf{h}_2) = \emptyset, \\
 & & \mathsf{h} = \mathsf{h}_1 \cup \mathsf{h}_2 \text{ and } \mathsf{h}_i \Vdash^{\nu}_{\Delta} \phi_i, \text{ for both } i = 1, 2
\end{array}$$

where $\mathrm{dom}(\mathsf{h}) \stackrel{\mathsf{def}}{=} \{c \in \mathbb{C} \mid \mathsf{h}(c) \text{ is defined}\}$ is the domain of the heap and (dis-)equalities, predicate atoms and existential quantifiers are defined same as for CL.

2.4 Decision Problems

We define the decision problems that are the focus of the upcoming sections. As usual, a decision problem is a class of yes/no queries that differ only in their input. In our case, the input consists of an SID and one or two predicates, written between square brackets.

Definition 6. *We consider the following problems, for a SID Δ and predicates $\mathsf{A}, \mathsf{B} \in \mathbb{A}$:*

1. $\mathsf{Sat}[\Delta, \mathsf{A}]$*: is the sentence* $\exists x_1 \ldots \exists x_{\#\mathsf{A}} \,.\, \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$ *satisfiable for* Δ*?*
2. $\mathsf{Tight}[\Delta, \mathsf{A}]$*: is every model* γ *of the sentence* $\exists x_1 \ldots \exists x_{\#\mathsf{A}} \,.\, \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$ *a tight configuration?*
3. $\mathsf{Bnd}[\Delta, \mathsf{A}]$*: is the set* $\{\delta(\gamma) \mid \gamma \models_{\Delta} \exists x_1 \ldots \exists x_{\#\mathsf{A}} \,.\, \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})\}$ *finite?*
4. $\mathsf{Entl}[\Delta, \mathsf{A}, \mathsf{B}]$*: does* $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \models_{\Delta} \exists x_{\#\mathsf{B}+1} \ldots \exists x_{\#\mathsf{A}} \,.\, \mathsf{B}(x_1, \ldots, x_{\#\mathsf{B}})$ *hold?*

We define the size of a formula ϕ as the total number of occurrences of symbols needed to write it down, denoted by $\mathrm{size}(\phi)$. The size of a SID Δ is $\mathrm{size}(\Delta) \stackrel{\mathsf{def}}{=} \sum_{\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \phi \in \Delta} \mathrm{size}(\phi) + \#\mathsf{A} + 1$. Other parameters of a SID Δ are its:

[2] We use the universe $\mathbb{C}$ here for simplicity, the definition works with any countably infinite set.

– *maximal arity*, denoted as $\mathrm{arity}(\Delta) \stackrel{\mathsf{def}}{=} \max\{\#\mathsf{A} \mid \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \leftarrow \phi \in \Delta\}$,
– *width*, denoted as $\mathrm{width}(\Delta) \stackrel{\mathsf{def}}{=} \max\{\mathrm{size}(\phi) \mid \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \leftarrow \phi \in \Delta\}$,
– *maximal interaction size*, denoted as $\mathrm{intersize}(\Delta) \stackrel{\mathsf{def}}{=} \max\{n \mid \langle x_1.p_1,\ldots,x_n.p_n\rangle$ occurs in ϕ, $\mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \leftarrow \phi \in \Delta\}$.

For each decision problem $\mathsf{P}[\Delta,\mathsf{A},\mathsf{B}]$, we consider its (k,ℓ)-bounded versions $\mathsf{P}^{(k,\ell)}[\Delta,\mathsf{A},\mathsf{B}]$, obtained by restricting the predicates and interaction atoms occurring Δ to $\mathrm{arity}(\Delta) \leq k$ and $\mathrm{intersize}(\Delta) \leq \ell$, respectively, where k and ℓ are either positive integers or infinity. We consider, for each $\mathsf{P}[\Delta,\mathsf{A},\mathsf{B}]$, the subproblems $\mathsf{P}^{(k,\ell)}[\Delta,\mathsf{A},\mathsf{B}]$ corresponding to the three cases (1) $k < \infty$ and $\ell = \infty$, (2) $k = \infty$ and $\ell < \infty$, and (3) $k = \infty$ and $\ell = \infty$. As we explain next, this is because, for the decision problems considered (Def. 6), the complexity for the case $k < \infty, \ell < \infty$ matches the one for the case $k < \infty, \ell = \infty$.

Moreover, for each problem $\mathsf{P}[\Delta,\mathsf{A}]$ (resp. $\mathsf{P}[\Delta,\mathsf{A},\mathsf{B}]$), we consider its general version $\mathsf{P}[\Delta,\phi]$ (resp. $\mathsf{P}[\Delta,\phi,\psi]$), where ϕ and ψ are CL formulæ, whose predicates are interpreted by the rules in Δ. The generalized problems $\mathsf{P}[\Delta,\phi]$ involving one predicate atom (points 1 and 3 of Def. 6) can be reduced to their restricted versions $\mathsf{P}[\Delta,\mathsf{A}]$, by introducing a fresh predicate A_ϕ (not occurring in Δ), of arity $n \geq 0$ and a rule $\mathsf{A}_\phi(x_1,\ldots,x_n) \leftarrow \phi$, where $\mathrm{fv}(\phi) = \{x_1,\ldots,x_n\}$. This reduction is linear in the size of the input and changes none of the following complexity results. Concerning the generalized entailment problem $\mathsf{Entl}[\Delta,\phi,\psi]$, the reduction to the problem $\mathsf{Entl}[\Delta,\mathsf{A},\mathsf{B}]$ (Def. 6 4) might affect its decidability status, which is subject to syntactic restrictions on the rules in Δ (details will be given in §6).

Satisfiability (1) and entailment (4) arise naturally during verification of reconfiguration programs. For instance, $\mathsf{Sat}[\Delta,\phi]$ asks whether a specification ϕ of a set configurations (e.g., a pre-, post-condition, or a loop invariant) is empty or not (e.g., an empty precondition typically denotes a vacuous verification condition), whereas $\mathsf{Entl}[\Delta,\phi,\psi]$ is used as a side condition for the Hoare rule of consequence, as in e.g., the proof from Fig. 1 (c). Moreover, entailments must be proved when checking inductiveness of a user-provided loop invariant.

In contrast, the applications of the tightness (2) and boundedness (3) problems are less obvious and require a few explanations. The $\mathsf{Tight}[\Delta,\phi]$ problem is relevant in the context of compositional verification of distributed systems. Suppose we have a distributed system consisting of two interacting subsystems, whose sets of initial configurations are described by ϕ_1 and ϕ_2, respectively i.e., the initial configurations of the system are described by $\phi_1 * \phi_2$. The compositional verification of a reconfiguration program P reduces checking the validity of a Hoare triple $\{\phi_1 * \phi_2\}\ \mathsf{P}\ \{\psi_1 * \psi_2\}$ to checking the validity of the simpler $\{\phi_i\}\ \mathsf{P}\ \{\psi_i\}$, for $i = 1,2$. Unfortunately, this appealing method faces the problem of *interference* between the subsystems described by ϕ_1 and ϕ_2, namely the loose interactions of ϕ_i might connect to present components of ϕ_{3-i} and change their states during the execution. In this case, it is sufficient to infer the sets of cross-boundary interactions $\mathcal{F}_{i,3-i}$, describing those interactions from ϕ_i that connect to components from ϕ_{3-i}, and check the validity of the triples $\{\phi_i * \mathcal{F}_{3-i,i}\}\ \mathsf{P}\ \{\psi_i * \mathcal{F}_{3-i,i}\}$, under a relaxed semantics which considers that the interactions in $\mathcal{F}_{3-i,i}$ can fire anytime, or according to the order described by some regular language. However, if $\mathsf{Tight}[\Delta,\phi_1]$ (resp. $\mathsf{Tight}[\Delta,\phi_2]$) has a negative answer, the set of cross-boundary interactions may be unbounded, hence not representable by a finite sep-

arating conjunction of interaction atoms $\mathcal{F}_{1,2}$ (resp. $\mathcal{F}_{2,1}$). Thus, the tightness problem is important in establishing necessary conditions under which a compositional proof rule can be applied to checking correctness of reconfigurations in a distributed system.

The $\mathsf{Bnd}[\Delta,\phi]$ problem is used to check a necessary condition for the decidability of entailments i.e., $\mathsf{Entl}[\Delta,\phi,\psi]$. If $\mathsf{Bnd}[\Delta,\phi]$ has a positive answer, we can reduce the problem $\mathsf{Entl}[\Delta,\phi,\psi]$ to an entailment problem for SL, which is always interpreted over heaps of bounded degree [20]. Otherwise, the decidability status of the entailment problem is open, for configurations of unbounded degree, such as the one described by the example below.

Example 2. The following SID describes star topologies with a central controller connected to an unbounded number of workers stations:

$$\mathit{Star}(x) \leftarrow [x] * \mathit{Worker}(x),\ \mathit{Worker}(x) \leftarrow \mathsf{emp} \mid \exists y\ .\ \langle x.\mathit{out}, y.\mathit{in}\rangle * [y] * \mathit{Worker}(x)\ \blacksquare$$

3 Satisfiability

We show that the satisfiability problem (Def. 6, point 1) is decidable, using a method similar to the one pioneered by Brotherston et al. [9], for checking satisfiability of inductively defined symbolic heaps in SL. We recall that a formula π is *pure* if and only if it is a separating conjunction of equalities, disequalities and state atoms.

Definition 7. *The* closure $\mathrm{cl}(\pi)$ *of a pure formula* π *is the limit of the sequence* $\pi^0, \pi^1, \pi^2, \ldots$ *such that* $\pi^0 = \pi$ *and, for each* $i \geq 0$, π^{i+1} *is obtained by joining (with* $*$*) all of the following formulæ to* π^i*:*

- $x = z$*, where* x *and* z *are the same variable, or* $x = y$ *and* $y = z$ *both occur in* π^i*,*
- $x \neq z$*, where* $x = y$ *and* $y \neq z$ *both occur in* π^i*, or*
- $y@q$*, where* $x@q$ *and* $x = y$ *both occur in* π^i*.*

Because only finitely many such formulæ can be added, the sequence of pure formulæ from Def. 7 is bound to stabilize after polynomially many steps. A pure formula is satisfiable if and only if its closure does not contain contradictory literals i.e., $x = y$ and $x \neq y$, or $x@q$ and $x@q'$, for $q \neq q' \in Q$. We write $x \approx_\pi y$ (resp. $x \not\approx_\pi y$) if and only if $x = y$ (resp. $x \neq y$) occurs in $\mathrm{cl}(\pi)$ and $\mathrm{not}(x \approx_\pi y)$ (resp. $\mathrm{not}(x \not\approx_\pi y)$) whenever $x \approx_\pi y$ (resp. $x \not\approx_\pi y$) does not hold. Note that e.g., $\mathrm{not}(x \approx_\pi y)$ is not the same as $x \not\approx_\pi y$.

Lemma 1. *A pure formula* π *is satisfiable if and only if the following hold:*

1. *for all* $x, y \in \mathrm{fv}(\pi)$*,* $x = y$ *and* $x \neq y$ *do not occur both in* $\mathrm{cl}(\pi)$*,*
2. *for all* $x \in \mathrm{fv}(\pi)$ *and* $q \neq r \in Q$*,* $x@q$ *and* $x@r$ *do not occur both in* $\mathrm{cl}(\pi)$*.*

Proof. A pure formula π is satisfiable if and only if there exists a store ν and a configuration $(\emptyset, \emptyset, \rho)$, such that $(\emptyset, \emptyset, \rho) \models^\nu \pi$. "$\Leftarrow$" It is easy to see that $\approx_\pi$ is an equivalence relation, for each pure formula π. Given any state map ρ, we define ν by assigning each equivalence class of $\approx_\pi$ a distinct component c, such that $\rho(c) = q$ if $y@q$ occurs in π, for a variable y in the class. By the conditions of the Lemma, ρ and ν are well defined and we have $(\emptyset, \emptyset, \rho) \models^\nu \pi$, by definition. "$\Rightarrow$" If $(\emptyset, \emptyset, \rho) \models^\nu \pi$ then $(\emptyset, \emptyset, \rho) \models^\nu \mathrm{cl}(\pi)$, because each additional formula in $\mathrm{cl}(\pi)$ is a logical consequence of π. Since $\mathrm{cl}(\pi)$ is satisfiable, the two conditions of the Lemma must hold. $\square$

Base tuples constitute the abstract domain used by the algorithms for checking satisfiability (point 1 of Def. 6) and boundedness (point 3 of Def. 6), defined as follows:

Definition 8. *A* base tuple *is a triple* $\mathfrak{t} = (\mathcal{C}^\sharp, I^\sharp, \pi)$*, where:*
- $\mathcal{C}^\sharp \in \mathrm{mpow}(\mathbb{V})$ *is a multiset of variables denoting present components,*
- $I^\sharp : \mathsf{Inter} \to \mathrm{mpow}(\mathbb{V}^+)$ *maps each interaction type* $\tau \in \mathsf{Inter}$ *into a multiset of tuples of variables of length* $|\tau|$ *each, and*
- π *is a pure formula.*

A base tuple is called satisfiable *if and only if* π *is satisfiable and the following hold:*
1. *for all* $x, y \in \mathcal{C}^\sharp$, $\mathrm{not}(x \approx_\pi y)$,
2. *for all* $\tau \in \mathsf{Inter}$, $\langle x_1, \ldots, x_{|\tau|}\rangle, \langle y_1, \ldots, y_{|\tau|}\rangle \in I^\sharp(\tau)$*, there exists* $i \in [1, |\tau|]$ *such that* $\mathrm{not}(x_i \approx_\pi y_i)$,
3. *for all* $\tau \in \mathsf{Inter}$, $\langle x_1, \ldots, x_{\#\tau}\rangle \in I^\sharp(\tau)$ *and* $1 \leq i < j \leq |\tau|$*, we have* $\mathrm{not}(x_i \approx_\pi x_j)$.

We denote by $\mathsf{SatBase}$ *the set of satisfiable base tuples.*

Note that a base tuple $(\mathcal{C}^\sharp, I^\sharp, \pi)$ is unsatisfiable if $\mathcal{C}^\sharp$ ($I^\sharp$) contains the same variable (tuple of variables) twice (for the same interaction type), hence the use of multisets in the definition of base tuples. It is easy to see that checking the satisfiability of a given base tuple $(\mathcal{C}^\sharp, I^\sharp, \pi)$ can be done in time $poly(\|\mathcal{C}^\sharp\| + \sum_{\tau \in \mathsf{Inter}} \|I^\sharp(\tau)\| + \mathrm{size}(\pi))$.

We define a partial *composition* operation on satisfiable base tuples, as follows:

$$(\mathcal{C}_1^\sharp, I_1^\sharp, \pi_1) \otimes (\mathcal{C}_2^\sharp, I_2^\sharp, \pi_2) \stackrel{\mathsf{def}}{=} (\mathcal{C}_1^\sharp \cup \mathcal{C}_2^\sharp, I_1^\sharp \cup I_2^\sharp, \pi_1 * \pi_2)$$

where the union of multisets is lifted to functions $\mathsf{Inter} \to \mathrm{mpow}(\mathbb{V}^+)$ in the usual way. The composition operation $\otimes$ is undefined if $(\mathcal{C}_1^\sharp, I_1^\sharp, \pi_1) \otimes (\mathcal{C}_2^\sharp, I_2^\sharp, \pi_2)$ is not satisfiable e.g., if $\mathcal{C}_1^\sharp \cap \mathcal{C}_2^\sharp \neq \emptyset$, $I_1^\sharp(\tau) \cap I_2^\sharp(\tau) \neq \emptyset$, for some $\tau \in \mathsf{Inter}$, or $\pi_1 * \pi_2$ is not satisfiable.

Given a pure formula π and a set of variables X, the projection $\pi{\downarrow_X}$ removes from π all atomic propositions α, such that $\mathrm{fv}(\alpha) \not\subseteq X$. The *projection* of a base tuple $(\mathcal{C}^\sharp, I^\sharp, \pi)$ on a variable set X is formally defined below:

$$(\mathcal{C}^\sharp, I^\sharp, \pi){\downarrow_X} \stackrel{\mathsf{def}}{=} \left(\mathcal{C}^\sharp \cap X, \lambda\tau \,.\, \{\langle x_1, \ldots, x_{|\tau|}\rangle \in I^\sharp(\tau) \mid x_1, \ldots, x_{|\tau|} \in X\}, \mathrm{cl}(\mathrm{dist}(I^\sharp) * \pi){\downarrow_X}\right)$$

$$\text{where } \mathrm{dist}(I^\sharp) \stackrel{\mathsf{def}}{=} \mathbin{\ast}_{\tau \in \mathsf{Inter}} \mathbin{\ast}_{\langle x_1, \ldots, x_{|\tau|}\rangle \in I^\sharp(\tau)} \mathbin{\ast}_{1 \leq i < j \leq |\tau|} x_i \neq x_j$$

The *substitution* operation $(\mathcal{C}^\sharp, I^\sharp, \pi)[x_1/y_1, \ldots, x_n/y_n]$ replaces simultaneously each x_i with y_i in $\mathcal{C}^\sharp$, $I^\sharp$ and π, respectively. For a store ν, we denote by $\nu[x_1/y_1, \ldots, x_n/y_n]$ the store such that $\nu[x_1/y_1, \ldots, x_n/y_n](x_i) = \nu(y_i)$ and agrees with ν over $\mathbb{V} \setminus \{x_1, \ldots, x_n\}$.We lift the composition, projection and substitution operations to sets of satisfiable base tuples, as usual.

Lemma 2. *Given a formula* ϕ *and a substitution* $\sigma = [x_1/y_1, \ldots, x_n/y_n]$*, for any configuration* (C, I, ρ) *and store* ν*,* $(C, I, \rho) \models^\nu_\Delta \phi\sigma$ *only if* $(C, I, \rho) \models^{\nu\sigma}_\Delta \phi$.

Proof. By induction on the definition of $\models^\nu_\Delta$. □

Next, we define the base tuple corresponding to a quantifier- and predicate-free formula $\phi = \psi * \pi$, where ψ consists of component and interaction atoms and π is pure. Since, moreover, we are interested in those components and interactions that are visible

input: a SID Δ
output: $\mu\overrightarrow{\mathcal{X}}.\Delta^\sharp$

1: initially $\mu\overrightarrow{\mathcal{X}}.\Delta^\sharp := \lambda \mathsf{A} \,.\, \emptyset$
2: **for** $\mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi \in \Delta$, with ϕ quantifier- and predicate-free **do**
3: $\quad \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{A}) := \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{A}) \cup \mathsf{Base}(\phi, \{x_1,\ldots,x_{\#\mathsf{A}}\})\downarrow_{x_1,\ldots,x_{\#\mathsf{A}}}$
4: **while** $\mu\overrightarrow{\mathcal{X}}.\Delta^\sharp$ still change **do**
5: $\quad$ **for** $\mathsf{r} : \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \divideontimes_{\ell=1}^{h} \mathsf{B}_\ell(z_1^\ell,\ldots,z_{\#\mathsf{B}_\ell}^\ell) \in \Delta$ **do**
6: $\quad\quad$ **if** there exist $\mathsf{t}_1 \in \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{B}_1),\ldots,\mathsf{t}_h \in \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{B}_h)$ **then**
7: $\quad\quad\quad \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{A}) := \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{A}) \cup \left(\mathsf{Base}(\phi,\{x_1,\ldots,x_{\#\mathsf{A}}\}) \otimes \bigotimes_{\ell=1}^{h} \mathsf{t}_\ell[x_1/z_1^\ell,\ldots,x_{\#\mathsf{B}_\ell}/z_{\#\mathsf{B}_\ell}^\ell]\right)\downarrow_{x_1,\ldots,x_{\#\mathsf{A}}}$

Fig. 2: Algorithm for the Computation of the Least Solution

through a given indexed set of parameters $X = \{x_1,\ldots,x_n\}$, for a variable y, we denote by $\{\{y\}\}_\pi^X$ the parameter x_i with the least index, such that $y \approx_\pi x_i$, or y itself, if no such parameter exists. We define the following sets of formulæ:

$$\mathsf{Base}(\phi,X) \stackrel{\text{def}}{=} \begin{cases} \{(\mathcal{C}^\sharp, I^\sharp, \pi)\} & \text{, if } (\mathcal{C}^\sharp, I^\sharp, \pi) \text{ is satisfiable} \\ \emptyset & \text{, otherwise} \end{cases}$$

$$\text{where } \mathcal{C}^\sharp \stackrel{\text{def}}{=} \{\{\{x\}\}_\pi^X \mid [x] \text{ occurs in } \psi\}$$

$$I^\sharp \stackrel{\text{def}}{=} \lambda\langle p_1,\ldots,p_s\rangle.\left\{\left\langle \{\{y_1\}\}_\pi^X,\ldots,\{\{y_s\}\}_\pi^X\right\rangle \mid \langle y_1.p_1,\ldots,y_s.p_s\rangle \text{ occurs in } \psi\right\}$$

We consider a tuple of variables $\overrightarrow{\mathcal{X}}$, having a variable $\mathcal{X}(\mathsf{A})$ ranging over $\text{pow}(\mathsf{SatBase})$, for each predicate A that occurs in Δ. With these definitions, each rule of Δ:

$$\mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \mathsf{B}_1(z_1^1,\ldots,z_{\#\mathsf{B}_1}^1) * \ldots * \mathsf{B}_h(z_1^h,\ldots,z_{\#\mathsf{B}_h}^h)$$

where ϕ is a quantifier- and predicate-free formula, induces the constraint:

$$\mathcal{X}(\mathsf{A}) \supseteq \left(\mathsf{Base}(\phi,\{x_1,\ldots,x_{\#\mathsf{A}}\}) \otimes \bigotimes_{\ell=1}^{h} \mathcal{X}(\mathsf{B}_\ell)[x_1/z_1^\ell,\ldots,x_{\#\mathsf{B}_\ell}/z_{\#\mathsf{B}_\ell}^\ell]\right)\downarrow_{x_1,\ldots,x_{\#\mathsf{A}}} \qquad (1)$$

Let $\Delta^\sharp$ be the set of such constraints, corresponding to the rules in Δ and let $\mu\overrightarrow{\mathcal{X}}.\Delta^\sharp$ be the tuple of least solutions of the constraint system generated from Δ, indexed by the tuple of predicates that occur in Δ, such that $\mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{A})$ denotes the entry of $\mu\overrightarrow{\mathcal{X}}.\Delta^\sharp$ correponding to A. Since the composition and projection are monotonic operations, such a least solution exists and is unique. Moreover, since $\mathsf{SatBase}$ is finite, the least solution can be attained in a finite number of steps, using a standard Kleene iteration(see Fig. 2).

Given a base tuple $(\mathcal{C}^\sharp, I^\sharp, \pi)$ and a store ν, we define the following sets of components and interactions, respectively:

$$\nu(\mathcal{C}^\sharp) \stackrel{\text{def}}{=} \{\nu(x) \mid x \in \mathcal{C}^\sharp\}$$

$$\nu(I^\sharp) \stackrel{\text{def}}{=} \bigcup_{\langle p_1,\ldots,p_n\rangle \in \mathsf{Inter}} \{(\nu(x_1),p_1,\ldots,\nu(x_n),p_n) \mid (x_1,\ldots,x_n) \in I^\sharp(\langle p_1,\ldots,p_n\rangle)\}$$

We state below the main result leading to an elementary recursive algorithm for the satisfiability problem (Thm. 1).

Lemma 3. *Given a base tuple* $(\mathcal{C}^\sharp, I^\sharp, \pi) \in \mu\overrightarrow{X}.\Delta^\sharp(\mathsf{A})[x_1/y_1, \ldots, x_{\#\mathsf{A}}/y_{\#\mathsf{A}}]$*, a state map* ρ *and a store* ν *such that* $(\emptyset, \emptyset, \rho) \models^\nu \pi$*, a set of components* $\mathcal{D}$ *disjoint from* $\nu(\mathcal{C}^\sharp)$ *and a set of interactions* $\mathcal{J}$ *disjoint from* $\nu(I^\sharp)$*, there exists a configuration* $(\mathcal{C}, I, \rho)$*, such that* $(\mathcal{C}, I, \rho) \models^\nu_\Delta \mathsf{A}(y_1, \ldots, y_{\#\mathsf{A}})$*,* $\mathcal{C} \cap \mathcal{D} = \emptyset$ *and* $I \cap \mathcal{J} = \emptyset$*.*

Proof. Let $\sigma \stackrel{\text{def}}{=} [x_1/y_1, \ldots, x_{\#\mathsf{A}}/y_{\#\mathsf{A}}]$ be a substitution and $(\mathcal{C}_0^\sharp, I_0^\sharp, \pi_0) \in \mu\overrightarrow{X}.\Delta^\sharp(\mathsf{A})$ be a base pair, such that $(\mathcal{C}^\sharp, I^\sharp, \pi) = (\mathcal{C}_0^\sharp, I_0^\sharp, \pi_0)\sigma$. Since $(\emptyset, \emptyset, \rho) \models^\nu \pi$, we obtain $(\emptyset, \emptyset, \rho) \models^{\nu_0} \pi_0$, by Lemma 2, where we define $\nu_0 \stackrel{\text{def}}{=} \nu\sigma$. Let

$$\mathcal{K} \stackrel{\text{def}}{=} \mathcal{D} \cup \{c_i \mid (c_1, p_1, \ldots, c_n, p_n) \in \mathcal{J},\ i \in [1, n]\}$$

The proof is by fixpoint induction on the definition of $(\mathcal{C}_0^\sharp, I_0^\sharp, \pi_0)$. Assume that:

$$(\mathcal{C}_0^\sharp, I_0^\sharp, \pi_0) \in \big(\mathsf{Base}(\psi * \pi', \{x_1, \ldots, x_{\#\mathsf{A}}\}) \otimes \bigotimes_{\ell=2}^{h} \mu\overrightarrow{X}.\Delta^\sharp(\mathsf{B}_\ell)[x_1/z_1^\ell, \ldots, x_{\#\mathsf{B}_\ell}/z_{\#\mathsf{B}_\ell}^\ell]\big)\downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}$$

for a rule $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \psi * \pi' * \circledast_{\ell=2}^{h} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$ of Δ, such that $\psi * \pi'$ is quantifier-free, ψ consists of component and interaction atoms and π' is the largest pure subformula of $\psi * \pi'$. Then there exist base tuples $(\mathcal{C}_1^\sharp, I_1^\sharp, \pi_1), \ldots, (\mathcal{C}_h^\sharp, I_h^\sharp, \pi_h)$, such that:

- $(\mathcal{C}_1^\sharp, I_1^\sharp, \pi_1) \in \mathsf{Base}(\psi * \pi', \{x_1, \ldots, x_{\#\mathsf{A}}\})$,
- $(\mathcal{C}_\ell^\sharp, I_\ell^\sharp, \pi_\ell) \in \mu\overrightarrow{X}.\Delta^\sharp(\mathsf{B}_\ell)[x_1/z_1^\ell, \ldots, x_{\#\mathsf{B}_\ell}/z_{\#\mathsf{B}_\ell}^\ell]$, for all $\ell \in [2, h]$,
- $(\mathcal{C}_0^\sharp, I_0^\sharp, \pi_0) = \big((\mathcal{C}_1^\sharp, I_1^\sharp, \pi_1) \otimes \ldots \otimes (\mathcal{C}_h^\sharp, I_h^\sharp, \pi_h)\big)\downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}$.

From the first and last points, we deduce $\pi_0 = \mathrm{cl}(\pi' * \circledast_{\ell=2}^{h}(\mathrm{dist}(I_\ell^\sharp) * \pi_\ell))\downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}$. Let $\pi'' \stackrel{\text{def}}{=} \pi' * \circledast_{\ell=2}^{h}(\mathrm{dist}(I_\ell^\sharp) * \pi_\ell)$ and define a store ν_0', by assigning each $\approx_{\pi''}$-equivalence class the following component:

- $\nu_0(x_i)$, if x_i belongs to the class, for some $i \in [1, \#\mathsf{A}]$,
- else, if the class is disjoint from $\{x_1, \ldots, x_{\#\mathsf{A}}\}$ and $y@q$ occurs in π', for a variable y in the class, we assign $c \in \mathbb{C} \setminus \mathcal{K}$, such that $\rho(c) = q$; since π' is satisfiable, there are no two state atoms $y@q$ and $z@r$, such that $y \approx_{\pi'} z$ and $q \neq r$ in π' and, moreover, chosing c is always possible, by the last point of Def. 1,
- otherwise, the class is assigned an arbitrary component $c \in \mathbb{C} \setminus \mathcal{K}$.

Such a store exists, because π'' is satisfiable and, moreover, $(\emptyset, \emptyset, \rho) \models^{\nu_0'} \pi''$, hence also $(\emptyset, \emptyset, \rho) \models^{\nu_0'} \pi_\ell$, for all $\ell \in [2, h]$. We define two sequences of sets of components $\mathcal{C}_1, \ldots, \mathcal{C}_h$ and interactions $I_1, \ldots, I_h$, as follows:

- $\mathcal{C}_1 \stackrel{\text{def}}{=} \{\nu_0'(y) \mid [y] \text{ occurs in } \psi\}$,
- $I_1 \stackrel{\text{def}}{=} \{(\nu_0'(z_1), p_1, \ldots, \nu_0'(z_t), p_t) \mid \langle z_1.p_1, \ldots, z_t.p_t\rangle \text{ occurs in } \psi\}$,
- for all $\ell \in [2, h]$, assume that $\mathcal{C}_1, \ldots, \mathcal{C}_{\ell-1}$ and $I_1, \ldots, I_{\ell-1}$ have been defined and let us define:

$$\mathcal{D}_\ell \stackrel{\text{def}}{=} \mathcal{D} \cup \bigcup_{j=1}^{\ell-1} \mathcal{C}_j \cup \bigcup_{j=\ell+1}^{h} \nu_0'(\mathcal{C}_j^\sharp)$$
$$\mathcal{J}_\ell \stackrel{\text{def}}{=} \mathcal{J} \cup \bigcup_{j=1}^{\ell-1} I_j \cup \bigcup_{j=\ell+1}^{h} \nu_0'(I_j^\sharp)$$

First, we prove that $\mathcal{D}_\ell \cap \nu'_0(\mathcal{C}^\sharp_\ell) = \emptyset$ and $\mathcal{J}_\ell \cap \nu'_0(I^\sharp_\ell) = \emptyset$ (we prove the first point, the second using a similar argument). Suppose, for a contradiction, that $c \in \mathcal{D}_\ell \cap \nu'_0(\mathcal{C}^\sharp_\ell)$. We distinguish the following cases:

- if $c \in \mathcal{D} \cap \nu'_0(\mathcal{C}^\sharp_\ell)$, then $c \in \mathcal{D} \cap \nu'_0(\mathcal{C}^\sharp_0)$, because $\mathcal{C}^\sharp_\ell \subseteq \mathcal{C}^\sharp_0$, contradiction with $\mathcal{D} \cap \nu'_0(\mathcal{C}^\sharp_0) = \mathcal{D} \cap \nu(\mathcal{C}^\sharp) = \emptyset$.
- else, if $c \in \mathcal{C}_j \cap \nu'_0(\mathcal{C}^\sharp_\ell)$, for some $j \in [1, \ell-1]$, then $c \in \mathcal{C}_j \cap \mathcal{D}_j$, because $\nu'_0(\mathcal{C}^\sharp_\ell) \subseteq \mathcal{D}_j$, contradiction with the inductive hypothesis $\mathcal{C}_j \cap \mathcal{D}_j = \emptyset$.
- otherwise, $c \in \nu'_0(\mathcal{C}^\sharp_j) \cap \nu'_0(\mathcal{C}^\sharp_\ell)$, hence there exist variables $y_j \in \mathcal{C}^\sharp_j$ and $y_\ell \in \mathcal{C}^\sharp_\ell$, such that $y_j \approx_{\pi''} y_\ell$, contradiction with the fact that $(\mathcal{C}^\sharp_j, I^\sharp_j, \pi_j) \otimes (\mathcal{C}^\sharp_\ell, I^\sharp_\ell, \pi_\ell)$ is satisfiable.

Second, we apply the inductive hypothesis to obtain configurations $(\mathcal{C}_\ell, I_\ell, \rho)$, such that $(\mathcal{C}_\ell, I_\ell, \rho) \models^{\nu'_0}_\Delta \mathsf{B}_\ell(t^\ell_1, \ldots, t^\ell_{\#\mathsf{B}_\ell})$, $\mathcal{C}_\ell \cap \mathcal{D}_\ell = \emptyset$ and $I_\ell \cap \mathcal{J}_\ell = \emptyset$, for all $\ell \in [2,h]$. By the definitions of $\mathcal{D}_\ell$ and $\mathcal{J}_\ell$, the sets $\mathcal{C}_\ell$ and I_ℓ are pairwise disjoint, respectively, hence the composition $(\mathcal{C}, I, \rho) \stackrel{\text{def}}{=} \bullet^h_{\ell=1}(\mathcal{C}_\ell, I_\ell, \rho)$ is defined. Moreover, $(\mathcal{C}, I, \rho) \models^{\nu'_0}_\Delta \psi * \pi' * \mathbin{⫽}^h_{\ell=2} \mathsf{B}_\ell(t^\ell_1, \ldots, t^\ell_{\#\mathsf{B}_\ell})$, hence $(\mathcal{C}, I, \rho) \models^{\nu'_0}_\Delta \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$, leading to $(\mathcal{C}, I, \rho) \models^{\nu} \mathsf{A}(y_1, \ldots, y_{\#\mathsf{A}})$.

Finally, we are left with proving that $\mathcal{C} \cap \mathcal{D} = \emptyset$ and $I \cap \mathcal{J} = \emptyset$ (we prove the first point only, the second uses a similar reasoning). Since $\mathcal{C} = \bigcup^h_{\ell=1} \mathcal{C}_\ell$, this is equivalent to proving the following:

- $\mathcal{C}_1 \cap \mathcal{D} = \emptyset$: suppose, for a contradiction, that $\nu'_0(y) \in \mathcal{D}$, for a variable y, such that $[y]$ occurs in ψ. By the definition of ν'_0, we have either $\nu'_0(y) \in \nu'_0(\mathcal{C}^\sharp_0)$ or $\nu'_0(y) \in \mathcal{K}$. Since $\nu'_0(\mathcal{C}^\sharp_0) \cap \mathcal{D} = \mathcal{K} \cap \mathcal{D} = \emptyset$, both cases lead to a contradiction.
- $\mathcal{C}_\ell \cap \mathcal{D} = \emptyset$, for all $\ell \in [2,h]$: because $\mathcal{C}_\ell \cap \mathcal{D}_\ell = \emptyset$ and $\mathcal{D} \subseteq \mathcal{D}_\ell$, by definition of $\mathcal{D}_\ell$, for all $\ell \in [2,h]$. □

Lemma 4. *Given a predicate atom* $\mathsf{A}(y_1, \ldots, y_{\#\mathsf{A}})$*, a store* ν *and a configuration* $(\mathcal{C}, I, \rho)$*, such that* $(\mathcal{C}, I, \rho) \models^{\nu}_\Delta \mathsf{A}(y_1, \ldots, y_{\#\mathsf{A}})$*, there exists a base tuple* $(\mathcal{C}^\sharp, I^\sharp, \pi) \in \mu\overrightarrow{X}.\Delta^\sharp(\mathsf{A})[x_1/y_1, \ldots, x_{\#\mathsf{A}}/y_{\#\mathsf{A}}]$*, such that* $\nu(\mathcal{C}^\sharp) \subseteq \mathcal{C}$*,* $\nu(I^\sharp) \subseteq I$ *and* $(\emptyset, \emptyset, \rho) \models^{\nu} \pi$*.*

Proof. By fixpoint induction on the definition of the satisfaction relation $\models^{\nu}_\Delta$. Since $(\mathcal{C}, I, \rho) \models^{\nu}_\Delta \mathsf{A}(y_1, \ldots, y_{\#\mathsf{A}})$, by Lemma 2, we have $(\mathcal{C}, I, \rho) \models^{\nu_0}_\Delta \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$, where $\nu_0 \stackrel{\text{def}}{=} \nu[x_1/y_1, \ldots, x_{\#\mathsf{A}}/y_{\#\mathsf{A}}]$. Hence, Δ has a rule $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \psi * \pi' * \mathbin{⫽}^h_{\ell=2} \mathsf{B}_\ell(z^\ell_1, \ldots, z^\ell_{\#\mathsf{B}_\ell})$, such that $\psi * \pi'$ is quantifier-free, ψ consists of component and interaction atoms and π' is pure and there exists a store ν'_0, that agrees with ν_0 over $x_1, \ldots, x_{\#\mathsf{A}}$ and configurations $(\mathcal{C}_1, I_1, \rho), \ldots, (\mathcal{C}_\ell, I_\ell, \rho)$, such that:

- $(\mathcal{C}_1, I_1, \rho) \models^{\nu'_0} \psi * \pi'$,
- $(\mathcal{C}_\ell, I_\ell, \rho) \models^{\nu'_0}_\Delta \mathsf{B}_\ell(z^\ell_1, \ldots, z^\ell_{\#\mathsf{B}_\ell})$, for all $\ell \in [2,h]$, and
- $(\mathcal{C}, I, \rho) = (\mathcal{C}_1, I_1, \rho) \bullet \ldots \bullet (\mathcal{C}_h, I_h, \rho)$.

We consider the following base tuples:

- $(\mathcal{C}^\sharp_1, I^\sharp_1, \pi_1) \stackrel{\text{def}}{=} \mathsf{Base}(\psi * \pi', \{x_1, \ldots, x_{\#\mathsf{A}}\})$,

– $(\mathcal{C}^\sharp_\ell, I^\sharp_\ell, \pi_\ell) \in \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{B}_\ell)[x_1/z^\ell_1, \ldots, x_{\#\mathsf{B}_\ell}/z^\ell_{\#\mathsf{B}_\ell}]$, such that $\nu'_0(\mathcal{C}^\sharp_\ell) \subseteq \mathcal{C}_\ell$, $\nu'_0(I^\sharp_\ell) \subseteq I_\ell$ and $(\emptyset, \emptyset, \rho) \models^{\nu'_0} \pi_\ell$, whose existence is guaranteed by the inductive hypothesis, for all $\ell \in [2,h]$.

By the definition of $\mathsf{Base}(\psi * \pi', \{x_1, \ldots, x_{\#\mathsf{A}}\})$ and the fact that $(\mathcal{C}_1, I_1, \rho) \models^{\nu'_0} \psi * \pi'$, we obtain $\nu'_0(\mathcal{C}^\sharp_1) = \mathcal{C}_1$ and $\nu'_0(I^\sharp_1) = I_1$. Since the composition $\bullet^h_{\ell=1}(\mathcal{C}_\ell, I_\ell, \rho)$ is defined, the sets $\mathcal{C}_1, \ldots, \mathcal{C}_h$ and $I_1, \ldots, I_h$ are pairwise disjoint, respectively. Since $\nu'_0(\mathcal{C}^\sharp_\ell) \subseteq \mathcal{C}_\ell$ and $\nu'_0(I^\sharp) \subseteq I_\ell$, for all $\ell \in [1,h]$, we deduce that $\bigotimes^h_{\ell=1}(\mathcal{C}^\sharp_\ell, I^\sharp_\ell, \pi_\ell)$ is satisfiable, because:

– for all $1 \leq i < j \leq h$, for any two variables $y \in \mathcal{C}^\sharp_i$ and $z \in \mathcal{C}^\sharp_j$ we have $\mathrm{not}(y \approx_{\pi'} z)$, because $\nu'_0(\mathcal{C}^\sharp_i) \cap \nu'_0(\mathcal{C}^\sharp_j) = \emptyset$,
– for all $1 \leq i < j \leq h$, all $\tau \in \mathsf{Inter}$, for any two tuples $\langle y_1, \ldots, y_{|\tau|} \rangle \in I^\sharp_i(\tau)$ and $\langle z_1, \ldots, z_{|\tau|} \rangle \in I^\sharp_j(\tau)$, we have $\mathrm{not}(y_k \approx_{\pi'} z_k)$, for at least some $k \in [1, |\tau|]$, because $\nu'_0(I^\sharp_i) \cap \nu'_0(I^\sharp_j) = \emptyset$,
– for each tuple $\langle y_1, \ldots, y_{|\tau|} \rangle \in I^\sharp_\ell(\langle p_1, \ldots, p_n \rangle)$, for $\ell \in [1,h]$, we have $\mathrm{not}(y_i \approx_{\pi'} y_j)$, for all $1 \leq i < j \leq n$, because $(\nu'_0(y_1), p_1, \ldots, \nu'_0(y_n), p_n) \in I_\ell$, hence $\nu'_0(y_1), \ldots, \nu'_0(y_n)$ are pairwise distinct,
– $(\emptyset, \emptyset, \rho) \models^{\nu'_0} \pi' * \circledast^h_{\ell=2} \pi_\ell$, hence $(\emptyset, \emptyset, \rho) \models^{\nu'_0} \pi' * \circledast^h_{\ell=2} \mathrm{dist}(I^\sharp_\ell) * \pi_\ell$, by the previous point.

Then we define $(\mathcal{C}^\sharp_0, I^\sharp_0, \pi_0) \stackrel{\mathsf{def}}{=} \left(\bigotimes^h_{\ell=1}(\mathcal{C}^\sharp_\ell, I^\sharp_\ell, \pi_\ell)\right)\downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}$ and $(\mathcal{C}^\sharp, I^\sharp, \pi) \stackrel{\mathsf{def}}{=} (\mathcal{C}^\sharp_0, I^\sharp_0, \pi_0)[x_1/y_1, \ldots, x_{\#\mathsf{A}}/y_{\#\mathsf{A}}]$. By the definition of $\Delta^\sharp$, we have:

$$\mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{A}) \supseteq \left(\mathsf{Base}(\psi * \pi', \{x_1, \ldots, x_{\#\mathsf{A}}\}) \otimes \bigotimes_{\ell=2}^{h} \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{B}_\ell)[x_1/z^\ell_1, \ldots, x_{\#\mathsf{B}_\ell}/z^\ell_{\#\mathsf{B}_\ell}]\right)\downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}$$

and, since, by the construction of $(\mathcal{C}^\sharp_0, I^\sharp_0, \pi_0)$,

$$(\mathcal{C}^\sharp_0, I^\sharp_0, \pi_0) \in \left(\mathsf{Base}(\psi * \pi', \{x_1, \ldots, x_{\#\mathsf{A}}\}) \otimes \bigotimes_{\ell=2}^{h} \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{B}_\ell)[x_1/z^\ell_1, \ldots, x_{\#\mathsf{B}_\ell}/z^\ell_{\#\mathsf{B}_\ell}]\right)\downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}$$

we obtain $(\mathcal{C}^\sharp_0, I^\sharp_0, \pi_0) \in \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{A})$, leading to $(\mathcal{C}^\sharp, I^\sharp, \pi) \in \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{A})[x_1/y_1, \ldots, x_{\#\mathsf{A}}/y_{\#\mathsf{A}}]$. Next, we check that $\nu(\mathcal{C}^\sharp) \subseteq \bigcup^h_{\ell=1} \nu'_0(\mathcal{C}^\sharp_\ell) \subseteq \bigcup^h_{\ell=1} \mathcal{C}_\ell = \mathcal{C}$ and $\nu(I^\sharp) \subseteq \bigcup^h_{\ell=1} \nu'_0(I^\sharp_\ell) \subseteq \bigcup^h_{\ell=1} I_\ell = I$. Finally, the requirement $(\emptyset, \emptyset, \rho) \models^\nu \pi$ follows from the following:

– $\pi = \pi_0[x_1/y_1, \ldots, x_{\#\mathsf{A}}/y_{\#\mathsf{A}}]$, by the definition of $(\mathcal{C}^\sharp, I^\sharp, \pi)$,
– $(\emptyset, \emptyset, \rho) \models^{\nu'_0} \pi'$ and $(\emptyset, \emptyset, \rho) \models^{\nu'_0} \pi_\ell$, for all $\ell \in [2,h]$,
– $\pi_0 = \mathrm{cl}(\pi' * \circledast^h_{\ell=2} \mathrm{dist}(I^\sharp_\ell) * \pi_\ell)\downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}$, where $(\emptyset, \emptyset, \rho) \models^{\nu'_0} \mathrm{dist}(I^\sharp_\ell)$ follows from the satisfiability of $(\mathcal{C}^\sharp_\ell, I^\sharp_\ell, \pi_\ell)$, for all $\ell \in [2,h]$. □

Lemma 5. $\mathsf{Sat}[\Delta, \mathsf{A}]$ *has a positive answer if and only if* $\mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{A}) \neq \emptyset$.

Proof. "⇐" follows from Lemma 3 and "⇒" follows from Lemma 4. □

If the maximal arity of the predicates occurring in Δ is bound by a constant k, no satisfiable base tuple $(\mathcal{C}^\sharp, I^\sharp, \pi)$ can have a tuple $\langle y_1, \ldots, y_{|\tau|}\rangle \in I^\sharp(\tau)$, for some $\tau \in \mathsf{Inter}$, such that $|\tau| > k$, since all variables $y_1, \ldots, y_{|\tau|}$ are parameters denoting distinct components (point 3 of Def. 8). Hence, the upper bound on the size of a satisfiable base tuple is constant, in both the $k < \infty, \ell < \infty$ and $k < \infty, \ell = \infty$ cases, which are, moreover indistinguishable complexity-wise (i.e., both are NP-complete). In contrast, in the cases $k = \infty, \ell < \infty$ and $k = \infty, \ell = \infty$, the upper bound on the size of satisfiable base tuples is polynomial and simply exponential in size(Δ), incurring a complexity gap of one and two exponentials, respectively. The theorem below states the main result of this section:

Theorem 1. $\mathsf{Sat}^{(k,\infty)}[\Delta, \mathsf{A}]$ *is* NP*-complete for* $k \geq 4$, $\mathsf{Sat}^{(\infty,\ell)}[\Delta, \mathsf{A}]$ *is* EXP*-complete and* $\mathsf{Sat}[\Delta, \mathsf{A}]$ *is in* 2EXP.

Proof. Membership (upper bounds). For non-negative integers $m \leq n$ denote by $S_{n,m} \stackrel{\text{def}}{=} \frac{n!}{(n-m)!}$ the number of ordered m-element subsets of a n-element set.

Let $\alpha = \text{arity}(\Delta)$, $\beta = \text{intersize}(\Delta)$ and $p = \|\mathcal{P}\|$. The maximum length of a satisfiable base tuple is $B \stackrel{\text{def}}{=} \alpha + (\sum_{j=1}^{\min(\alpha,\beta)} p^j \cdot S_{\alpha,j}) + (2\alpha^2 + \alpha)$, that is, size of the set of components $\mathcal{C}^\sharp$ plus the size of the set of interactions $I^\sharp$ plus the length of the longest pure formula π. In general, for any non-negative integer j there exists at most p^j interaction types of arity j with ports from $\mathcal{P}$; moreover, for any such interaction type there exists at most $S_{\alpha,j}$ interactions relating distinct components from an α-element set. Moreover, no such interaction exists neither if $j > \alpha$ nor $j > \beta$.

For any $u \leq \alpha$ it holds that $\sum_{j=1}^{u} p^j \cdot S_{\alpha,j} \leq p^u \alpha^u$ (an easy check by induction on u). We use the inequality above with $u = \min(\alpha, \beta)$ and obtain that $B \leq 2\alpha + 2\alpha^2 + p^{\min(\alpha,\beta)}\alpha^{\min(\alpha,\beta)} \stackrel{\text{def}}{=} B^*$. We distinguish the three cases:

1. $k < \infty$, $\ell = \infty$: since $\alpha \leq k$ then α is constant and $B^* = O(1)$,
2. $k = \infty$, $\ell < \infty$: since $\beta \leq \ell$ and $\alpha = O(\text{size}(\Delta))$ then $B^* = poly(\text{size}(\Delta))$,
3. $k = \infty$, $\ell = \infty$: since $\alpha = O(\text{size}(\Delta))$ then $B^* = 2^{poly(\text{size}(\Delta))}$.

Let $N \stackrel{\text{def}}{=} 2^{B^*}$, that is, (an over-approximation of) the total number of base tuples. Clearly, N is constant in case (1) and respectively $2^{poly(\text{size}(\Delta))}$ and $2^{2^{poly(\text{size}(\Delta))}}$ in cases (2), (3). Let L be the number of predicates occuring in Δ and H be the maximum number of predicates used in a term in Δ. Let observe that both L and H are in general $O(\text{size}(\Delta))$. Then the least solution $\mu\overrightarrow{X}.\Delta^\sharp$ has at most N base tuples for each predicate, hence at most $L \cdot N$ base tuples. Furthermore, for each rule of Δ the time to check and/or produce the base tuple $(\mathcal{C}_0^\sharp, I_0^\sharp, \pi_0)$ with respect to the rule constraint (1) and given arguments $(\mathcal{C}_j^\sharp, I_j^\sharp, \pi_j)_{j=1,h}$ is polynomial $poly(B^*, \text{size}(\Delta)))$. That is, both composition and projection take at most $(H+1)B^* + \text{size}(\Delta)^3$ time as they need to process (union or scan) at most $H+1$ base tuples of length B^* each plus the closure of pure formula with at most size(Δ) variables.

1. $k < \infty$, $\ell = \infty$: We define a non-determinstic algorithm as follows. Let (Δ, A) be the input instance. We guess a witness $\langle W_1, \ldots, W_K\rangle$ for a least solution, where $1 \leq K \leq L \cdot N$ and each W_i entry is of the form $(T_i, r_i, e_{i,1}, \ldots, e_{i,h_i})$, where T_i is a base tuple, r_i an index of a rule of Δ and $e_{i,1}, \ldots, e_{i,h}$ are index values from $\{i+1, \ldots, K\}$ for $0 \leq h_i \leq H$. The length of every witness entry is therefore at

most $B^* + \lceil \log_2(\text{size}(\Delta)) \rceil + H \lceil \log_2(L \cdot N) \rceil$. As N is constant when $k < \infty$, and L and H are $O(\text{size}(\Delta))$ it follows that the number of guesses is polynomial for building $\langle W_1, \ldots, W_K \rangle$. We now check that $\langle W_1, \ldots, W_K \rangle$ represents indeed a valid computation of a base tuples from the least solution i.e., $(\mathcal{C}^\sharp, I^\sharp, \pi) \in \overrightarrow{\mu X}.\Delta^\sharp(\mathsf{A})$. For this, we need to check: (a) every entry is well-formed, that is, the rule indexed by r_i instantiates precisely h_i predicates; moreover, for every $1 \leq j \leq h_i$ the index $e_{i,j}$ designates an entry $W_{e_{i,j}}$ whose rule defines the j-th predicate instantiated by the rule r_i; (b) the base tuple of every entry is satisfiable and correctly computed, that is, T_i is the result of applying the constraint (1) for rule r_i with actual arguments $T_{e_{i,1}}, \ldots, T_{e_{i,h_i}}$ from the referred entries; (c) the rule r_1 of the first entry W_1 defines the predicate A. Again, as B^* and N are constant in this case, all these checks are done in polynomial time. Since both the generation and the checking of the witness are polynomial time, this ensures membership in NP.

2. $k = \infty$, $\ell < \infty$: Consider the computation of the least solution $\overrightarrow{\mu X}.\Delta^\sharp$ using standard Kleene iteration. At every step, a rule of Δ and a tuple of at most H base tuples arguments are selected to produce a new base tuple. Thus, in the worst case, at most $\text{size}(\Delta)$ rules in combination with at most N^H base tuples need to be selected and evaluated. If no new base tuple is generated the fixpoint is reached and the algorithms stops. Since there are at most $L \cdot N$ base tuples in the least solution, the total time will be therefore $L \cdot N \cdot \text{size}(\Delta) \cdot N^H \cdot t(B^*, \text{size}(\Delta))$ where $t(B^*, \text{size}(\Delta))$ is the (polynomial) time to process one selection. It is an easy check that the above is $2^{poly(\text{size}(\Delta))}$ since $N = 2^{poly(\text{size}(\Delta))}$ in this case.
3. $k = \infty$, $\ell = \infty$: Following the same reasoning as in the previous case the complexity is $2^{2^{poly(\text{size}(\Delta))}}$ as $N = 2^{2^{poly(\text{size}(\Delta))}}$ in this case.

Hardness (lower bounds). The restricted fragment of CL to $*, =, \neq$ is equisatisfiable to the restricted fragment of SL restricted to $*, =, \neq$. The satisfiability of the above SL fragment has been proven respectively NP-hard, if the arities of predicates are bounded by a constant $k \geq 3$ [9, Theorem 4.9] and EXP-hard, in general [9, Theorem 4.15]. Yet, the reductions considered in these proofs rely on the use of a predefined *nil* constant symbol in the SL logic; this constant can be nevertheless replaced by a variable consistently propagated along the SID, that is, at the price of increasing the arities of all predicates by one. Therefore, it follows immediately that $\mathsf{Sat}^{(k,\infty)}[\Delta, \phi]$ is NP-hard for $k \geq 4$ and $\mathsf{Sat}^{(\infty,\ell)}[\Delta, \phi]$, $\mathsf{Sat}[\Delta, \phi]$ are both EXP-hard. □

Example 3. The doubly-exponential upper bound for the algorithm computing the least solution of a system of constraints of the form (1) is necessary, in general, as illustrated by the following worst-case example. Let n be a fixed parameter and consider the n-arity predicates $A_1, \ldots, A_n$ defined by the following SID:

$$
\begin{array}{l}
A_i(x_1, \ldots, x_n) \rightarrow \circledast_{j=0}^{n-i} A_{i+1}(x_1, \ldots, x_{i-1}, [x_i, \ldots, x_n]^j) \text{ , for all } i \in [1, n-1] \\
A_n(x_1, \ldots, x_n) \rightarrow \langle x_1.p, \ldots, x_n.p \rangle \\
A_n(x_1, \ldots, x_n) \rightarrow \mathsf{emp}
\end{array}
$$

where, for a list of variables $x_i, \ldots, x_n$ and an integer $j \geq 0$, we write $[x_i, \ldots, x_n]^j$ for the list rotated to the left j times (e.g., $[x_1, x_2, x_3, x_4, x_5]^2 = x_3, x_4, x_5, x_1, x_2$). In this example, when starting with $A_1(x_1, \ldots, x_n)$ one eventually obtains predicate atoms $A_n(x_{i_1}, \ldots, x_{i_n})$,

for any permutation $x_{i_1},\ldots,x_{i_n}$ of $x_1,\ldots,x_n$. Since A_n may choose to create or not an interaction with that permutation of variables, the total number of base tuples generated for A_1 is $2^{n!}$. That is, the fixpoint iteration generates $2^{2^{O(n\log n)}}$ base tuples, whereas the size of the input of $\mathsf{Sat}[\Delta,\mathsf{A}]$ is $poly(n)$. ■

4 Tightness

The tightness problem (Def. 6, point 2) is the complement of a problem slightly stronger than satisfiability (1): given a SID Δ and a formula ϕ, such that $\mathrm{fv}(\phi)=\{x_1,\ldots,x_n\}$, the *looseness problem* $\mathsf{Loose}[\Delta,\mathsf{A}]$ asks for the existence of a loose configuration γ (Def. 2), such that $\gamma\models_\Delta \exists x_1\ldots\exists x_n\,.\,\phi$. We establish upper and lower bounds for the complexity of the looseness problem by a reduction to and from the satisfiability problem. The bounds for the tightness problem follow by standard complementation of the complexity classes for the looseness problem.

From Looseness to Satisfiability. Let Δ be a given SID and A be a predicate. For each predicate B that occurs in Δ, we consider a fresh predicate B', not occurring in Δ, such that $\#\mathsf{B}'=\#\mathsf{B}+1$. The SID $\widetilde{\Delta}$ consists of Δ and, for each rule of Δ of the form:

$$\mathsf{B}_0(x_1,\ldots,x_{\#\mathsf{B}_0})\leftarrow\exists y_1\ldots\exists y_m\,.\,\phi*\mathsf{B}_1(t_1^1,\ldots,t_{\#\mathsf{B}_1}^1)*\ldots*\mathsf{B}_h(t_1^h,\ldots,t_{\#\mathsf{B}_h}^h)$$

where ϕ is a quantifier- and predicate-free formula, $\widetilde{\Delta}$ has the following rules:

$$\mathsf{B}'_0(x_1,\ldots,x_{\#\mathsf{B}_0+1})\leftarrow\exists y_1\ldots\exists y_m\,.\,\phi*x_{\#\mathsf{B}_0+1}=z*\mathsf{B}_1(t_1^1,\ldots,t_{\#\mathsf{B}_1}^1)*\ldots*\mathsf{B}_h(t_1^h,\ldots,t_{\#\mathsf{B}_h}^h)$$

if z occurs in an interaction atom from ϕ, and:

$$\mathsf{B}'_0(x_1,\ldots,x_{\#\mathsf{B}_0+1})\leftarrow\exists y_1\ldots\exists y_m\,.\,\phi*\mathsf{B}_1(t_1^1,\ldots,t_{\#\mathsf{B}_1}^1)*\ldots*\mathsf{B}'_i(t_1^i,\ldots,t_{\#\mathsf{B}_i}^i,x_{\#\mathsf{B}_0+1})*\ldots*\mathsf{B}_h(t_1^h,\ldots,t_{\#\mathsf{B}_h}^h)$$

for some $i\in[1,h]$, if $h\geq 1$. Moreover, for each rule of Δ of the form above, with no predicate atoms (i.e., $h=0$), $\widetilde{\Delta}$ contains the rule:

$$\mathsf{B}'(x_1,\ldots,x_{\#\mathsf{B}+1})\leftarrow\exists y_1\ldots\exists y_m\,.\,\phi*[x_{\#\mathsf{B}_0+1}]$$

if and only if ϕ contains no predicate atoms. Finally, there is a fresh predicate $\widetilde{\mathsf{A}}$, of arity $\#\mathsf{A}$, with a rule:

$$\widetilde{\mathsf{A}}(x_1,\ldots,x_{\#\mathsf{A}})\leftarrow\exists y\,.\,\mathsf{A}'(x_1,\ldots,x_{\#\mathsf{A}},y)*[y]$$

Intuitively, the last parameter of a B' predicate binds to an arbitrary variable of an interaction atom. A configuration γ is loose if and only if the value (component) of some variable occurring in an interaction atom is absent, in which case the component can be added to γ (without clashing with a present component of γ) by the last rule. The reduction is polynomial, since the number of rules in $\widetilde{\Delta}$ is linear in the number of rules in Δ and the size of each newly added rule is increased by a constant. The following lemma states the correctness of the reduction:

Lemma 6. *Given a SID* Δ *and a predicate* A*, the problem* $\mathsf{Loose}[\Delta, \mathsf{A}]$ *has a positive answer if and only if the problem* $\mathsf{Sat}[\widetilde{\Delta}, \widetilde{\mathsf{A}}]$ *has a positive answer.*

Proof. "$\Rightarrow$" Let $\gamma \stackrel{\text{def}}{=} (\mathcal{C}, I, \rho)$ be a loose configuration, such that $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$, for some store ν. Since γ is loose, there exists an interaction $(c_1, p_1, \ldots, c_n, p_n) \in I$, such that $c_i \notin \mathcal{C}$, for some $i \in [1, n]$. We prove that $\gamma \models^{\nu[y \leftarrow c_i]}_{\widetilde{\Delta}} \mathsf{A}'(x_1, \ldots, x_{\#\mathsf{A}}, y)$, by fixpoint induction on the definition of $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$. This is sufficient, because then we obtain $(\mathcal{C} \cup \{c_i\}, I, \rho) \models^{\nu}_{\widetilde{\Delta}} \widetilde{\mathsf{A}}(x_1, \ldots, x_{\#\mathsf{A}})$, thus $\mathsf{Sat}[\widetilde{\Delta}, \widetilde{\mathsf{A}}]$ has a positive answer. Consider the rule of Δ:

$$\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \mathsf{B}_1(t^1_1, \ldots, t^1_{\#\mathsf{B}_1}) * \ldots * \mathsf{B}_h(t^h_1, \ldots, t^h_{\#\mathsf{B}_h})$$

where ϕ is quantifier- and predicate-free and $c'_1, \ldots, c'_m \in \mathbb{C}$ are components, such that $(\mathcal{C}, I, \rho) \models^{\nu'}_{\Delta} \phi * \mathsf{B}_1(t^1_1, \ldots, t^1_{\#\mathsf{B}_1}) * \ldots * \mathsf{B}_h(t^h_1, \ldots, t^h_{\#\mathsf{B}_h})$, where $\nu' \stackrel{\text{def}}{=} \nu[y_1 \leftarrow c'_1, \ldots, y_m \leftarrow c'_m]$. We distinguish the following cases:

- if $(c_1, p_1, \ldots, c_n, p_n) \in I$ because of an interaction atom $\langle z_1.p_1, \ldots, z_n.p_n \rangle$ from ϕ, such that $\nu'(z_i) = c_i$, for all $i \in [1, n]$, then $(\mathcal{C}, I, \rho) \models^{\nu'[y \leftarrow c_i]}_{\widetilde{\Delta}} \phi * y = z_i * \mathsf{B}_1(t^1_1, \ldots, t^1_{\#\mathsf{B}_1}) * \ldots * \mathsf{B}_h(t^h_1, \ldots, t^h_{\#\mathsf{B}_h})$, hence $(\mathcal{C}, I, \rho) \models^{\nu[y \leftarrow c_i]}_{\widetilde{\Delta}} \mathsf{A}'(x_1, \ldots, x_{\#\mathsf{A}}, y)$, by the definition of $\widetilde{\Delta}$.
- else $(c_1, p_1, \ldots, c_n, p_n) \in I$ because of a configuration γ', such that $\gamma = \gamma' \bullet \gamma''$, for some configuration γ'', and $\gamma' \models^{\nu'}_{\Delta} \mathsf{B}_i(t^i_1, \ldots, t^i_{\#\mathsf{B}_i})$. By the inductive hypothesis, we obtain $\gamma' \models^{\nu'[y \leftarrow c_i]}_{\Delta} \mathsf{B}'_i(t^i_1, \ldots, t^i_{\#\mathsf{B}_i}, y)$, hence $\gamma \models^{\nu[y \leftarrow c_i]}_{\Delta} \mathsf{A}'(x_1, \ldots, x_{\#\mathsf{A}}, y)$, because $\widetilde{\Delta}$ contains the rule:

$$\mathsf{A}'(x_1, \ldots, x_{\#\mathsf{A}+1}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \mathsf{B}_1(t^1_1, \ldots, t^1_{\#\mathsf{B}_1}) * \ldots * \mathsf{B}'_i(t^i_1, \ldots, t^i_{\#\mathsf{B}_i}, x_{\#\mathsf{A}+1}) * \ldots * \mathsf{B}_h(t^h_1, \ldots, t^h_{\#\mathsf{B}_h})$$

  and $\gamma'' \models^{\nu'} \phi * \mathsf{B}_1(t^1_1, \ldots, t^1_{\#\mathsf{B}_1}) * \ldots * \mathsf{B}'_{i-1}(t^{i-1}_1, \ldots, t^{i-1}_{\#\mathsf{B}_{i-1}}) * \mathsf{B}'_{i+1}(t^{i+1}_1, \ldots, t^{i+1}_{\#\mathsf{B}_{i+1}}) * \ldots * \mathsf{B}_h(t^h_1, \ldots, t^h_{\#\mathsf{B}_h})$ follows from $\gamma = \gamma' \bullet \gamma''$.

"$\Leftarrow$" Let $\gamma \stackrel{\text{def}}{=} (\mathcal{C}, I, \rho)$ be a configuration and ν be a store, such that $\gamma \models^{\nu}_{\widetilde{\Delta}} \widetilde{\mathsf{A}}(x_1, \ldots, x_{\#\mathsf{A}})$. Since the only rule of $\widetilde{\Delta}$ that defines $\widetilde{\mathsf{A}}$ is:

$$\widetilde{\mathsf{A}}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y \,.\, \mathsf{A}'(x_1, \ldots, x_{\#\mathsf{A}}, y) * [y]$$

there exists a component $c \in \mathcal{C}$, such that $(\mathcal{C} \setminus \{c\}, I, \rho) \models^{\nu[y \leftarrow c]}_{\widetilde{\Delta}} \mathsf{A}'(x_1, \ldots, x_{\#\mathsf{A}}, y)$. We prove the following:

- there exists an interaction $(c_1, p_1, \ldots, c_n, p_n) \in I$, such that $c_i = c$, and
- $(\mathcal{C} \setminus \{c\}, I, \rho) \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$,

by fixpoint induction on the definition of $(\mathcal{C} \setminus \{c\}, I, \rho) \models^{\nu[y \leftarrow c]}_{\widetilde{\Delta}} \mathsf{A}'(x_1, \ldots, x_{\#\mathsf{A}}, y)$. Based on the definition of $\widetilde{\Delta}$, we distinguish the following cases, where ϕ is quantifier- and predicate-free, $c'_1, \ldots, c'_m \in \mathbb{C}$ are components and $\nu' \stackrel{\text{def}}{=} \nu[y_1 \leftarrow c'_1, \ldots, y_m \leftarrow c'_m]$:

– $(\mathcal{C} \setminus \{c\}, I, \rho) \models^{\nu'[y \leftarrow c]}_{\widetilde{\Delta}} \phi * y = z * \mathsf{B}_1(t_1^1, \ldots, t_{\#\mathsf{B}_1}^1) * \ldots * \mathsf{B}_h(t_1^h, \ldots, t_{\#\mathsf{B}_h}^h)$, where z occurs in an interaction atom from ϕ. In this case, there exists an interaction $(c_1, p_1, \ldots, c_n, p_n) \in I$, such that $c = c_i$, for some $i \in [1,n]$. Moreover, $(\mathcal{C} \setminus \{c\}, I, \rho) \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$, because Δ has a rule:

$$\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \mathsf{B}_1(t_1^1, \ldots, t_{\#\mathsf{B}_1}^1) * \ldots * \mathsf{B}_h(t_1^h, \ldots, t_{\#\mathsf{B}_h}^h)$$

such that $(\mathcal{C} \setminus \{c\}, I, \rho) \models^{\nu}_{\Delta} \phi * \mathsf{B}_1(t_1^1, \ldots, t_{\#\mathsf{B}_1}^1) * \ldots * \mathsf{B}_h(t_1^h, \ldots, t_{\#\mathsf{B}_h}^h)$.

– $(\mathcal{C} \setminus \{c\}, I, \rho) \models^{\nu'[y \leftarrow c]}_{\widetilde{\Delta}} \phi * \mathsf{B}_1(t_1^1, \ldots, t_{\#\mathsf{B}_1}^1) * \ldots * \mathsf{B}'_i(t_1^i, \ldots, t_{\#\mathsf{B}_i}^i, y) * \ldots * \mathsf{B}_h(t_1^h, \ldots, t_{\#\mathsf{B}_h}^h)$. In this case, there exists configurations γ' and γ'', such that $(\mathcal{C} \setminus \{c\}, I, \rho) = \gamma' \bullet \gamma''$, $\gamma' \models^{\nu'[y \leftarrow c]}_{\widetilde{\Delta}} \mathsf{B}'_i(t_1^i, \ldots, t_{\#\mathsf{B}_i}^i, y)$ and $\gamma'' \models^{\nu}_{\Delta} \phi * \mathsf{B}_1(t_1^1, \ldots, t_{\#\mathsf{B}_1}^1) * \ldots * \mathsf{B}_h(t_1^h, \ldots, t_{\#\mathsf{B}_h}^h)$. By the inductive hypothesis, there exists an interaction $(c_1, p_1, \ldots, c_n, p_n)$ in γ', such that $c = c_i$, for some $i \in [1,n]$ and $\gamma' \models^{\nu}_{\Delta} \mathsf{B}_i(t_1^i, \ldots, t_{\#\mathsf{B}_i}^i)$. Then $(c_1, p_1, \ldots, c_n, p_n) \in I$ and $(\mathcal{C} \setminus \{c\}, I, \rho) \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$, since Δ has a rule:

$$\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \mathsf{B}_1(t_1^1, \ldots, t_{\#\mathsf{B}_1}^1) * \ldots * \mathsf{B}_h(t_1^h, \ldots, t_{\#\mathsf{B}_h}^h)$$

such that $\gamma' \bullet \gamma'' \models^{\nu'}_{\Delta} \phi * \mathsf{B}_1(t_1^1, \ldots, t_{\#\mathsf{B}_1}^1) * \ldots * \mathsf{B}_h(t_1^h, \ldots, t_{\#\mathsf{B}_h}^h)$.

– $(\mathcal{C} \setminus \{c\}, I, \rho) \models^{\nu'[y \leftarrow c]}_{\widetilde{\Delta}} \phi * [y]$ this case contradicts the semantics of CL. □

From Satisfiability to Looseness. Given a SID Δ and a predicate A, we build a SID $\widetilde{\Delta}$ that defines a predicate $\widetilde{\mathsf{A}}$, of equal arity, not occurring in Δ, such that $\mathsf{Sat}[\Delta, \mathsf{A}]$ has a positive answer if and only if there exists a loose configuration γ and a store ν, such that $\gamma \models^{\nu}_{\widetilde{\Delta}} \widetilde{\mathsf{A}}(x_1, \ldots, x_{\#\mathsf{A}})$. The rules of $\widetilde{\Delta}$ are the rules of Δ, to which the following rule is added, for some ports $p_1, p_2 \in \mathcal{P}$:

$$\widetilde{\mathsf{A}}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \exists y_2 \,.\, \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) * \langle y_1.p_1, y_2.p_2 \rangle$$

This reduction is polynomial, because we add one rule, of size linear on #A. The following lemma states the correctness of the reduction:

Lemma 7. *Given a SID Δ and a predicate A, the problem $\mathsf{Sat}[\Delta, \mathsf{A}]$ has a positive answer if and only if the problem $\mathsf{Loose}[\widetilde{\Delta}, \widetilde{\mathsf{A}}]$ has a positive answer.*

Proof. "⇒" If $\mathsf{Sat}[\Delta, \mathsf{A}]$ has a positive answer, there exists a configuration $\gamma \stackrel{\mathsf{def}}{=} (\mathcal{C}, I, \rho)$ and a store ν, such that $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$. Consider the configuration $\gamma' \stackrel{\mathsf{def}}{=} (\emptyset, \{(c_1, p_1, c_2, p_2)\}, \rho)$, for some components $c_1, c_2 \notin \mathcal{C}$. Then the composition $\gamma \bullet \gamma'$ is defined and we have $\gamma \bullet \gamma' \models^{\nu[y_1 \leftarrow c_1, y_2 \leftarrow c_2]}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) * \langle y_1.p_1, y_2.p_2 \rangle$, leading to $\gamma \bullet \gamma' \models^{\nu}_{\widetilde{\Delta}} \widetilde{\mathsf{A}}(x_1, \ldots, x_{\#\mathsf{A}})$. Moreover, $\gamma \bullet \gamma'$ is loose, because $c_1, c_2 \notin \mathcal{C}$. "⇐" If $\gamma \models^{\nu}_{\widetilde{\Delta}} \widetilde{\mathsf{A}}(x_1, \ldots, x_{\#\mathsf{A}})$, we necessarily have $\gamma \models^{\nu[y_1 \leftarrow c_1, y_2 \leftarrow c_2]}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) * \langle y_1.p_1, y_2.p_2 \rangle$, for some components $c_1, c_2 \in \mathbb{C}$, hence there exists a configuration γ', such that $\gamma' \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$. □

The polynomial reductions from Lemmas 6 and 7 establish the following complexity bounds for the tightness problem:

Theorem 2. $\mathsf{Tight}^{(k,\infty)}[\Delta,\mathsf{A}]$ *is* co-NP*-complete,* $\mathsf{Tight}^{(\infty,\ell)}[\Delta,\mathsf{A}]$ *is* EXP*-complete and* $\mathsf{Tight}[\Delta,\mathsf{A}]$ *is* 2EXP.

Proof. Since $\mathsf{Loose}^{(k,\infty)}[\Delta,\mathsf{A}]$ is polynomially-reducible to $\mathsf{Sat}^{(k+1,\infty)}[\Delta,\mathsf{A}]$, by Theorem 1, we obtain that $\mathsf{Loose}^{(k,\infty)}[\Delta,\mathsf{A}]$ is in NP. Moreover, since $\mathsf{Sat}^{(k,\infty)}[\Delta,\mathsf{A}]$ is polynomially-reducible to $\mathsf{Loose}^{(k,\infty)}[\Delta,\mathsf{A}]$, by Theorem 1, we obtain that $\mathsf{Loose}^{(k,\infty)}[\Delta,\mathsf{A}]$ is NP-complete. Because $\mathsf{Tight}^{(k,\infty)}[\Delta,\mathsf{A}]$ is the complement of $\mathsf{Loose}^{(k,\infty)}[\Delta,\mathsf{A}]$, we obtain that $\mathsf{Tight}^{(k,\infty)}[\Delta,\mathsf{A}]$ is co-NP-complete. The rest of the bounds are obtained by the same polynomial reductions and the fact that $\mathsf{Tight}^{(k,\infty)}[\Delta,\mathsf{A}]$ is the complement of $\mathsf{Loose}^{(k,\infty)}[\Delta,\mathsf{A}]$, for any k and ℓ, either integer constants, or infinity. □

5 Degree Boundedness

The boundedness problem (Def. 6, point 3) asks for the existence of a bound on the degree (Def. 4) of the models of a sentence $\exists x_1 \ldots \exists x_{\#\mathsf{A}} \,.\, \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}})$. For instance, it is possible to define inductively star topologies, with a central controller connected to an unbounded number of workers stations. Roughly speaking, the $\mathsf{Bnd}[\Delta,\mathsf{A}]$ problem has a negative answer if and only if there are increasingly large unfoldings (expansions of a formula by replacement of a predicate atom with one of its definitions) of $\mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}})$ repeating a rule that contains an interaction atom involving a parameter of the rule, which is always bound to the same component. For instance, the rule $\mathit{Worker}(x) \leftarrow \exists y \,.\, \langle x.\mathit{out}, y.\mathit{in}\rangle * [y] * \mathit{Worker}(x)$ (Example 2) declares an unbounded number of interactions $\langle x.\mathit{out}, y.\mathit{in}\rangle$ involving the component to which x is bound.

Definition 9. *Given a predicate* A *and a sequence* $(\mathsf{r}_1,i_1),\ldots,(\mathsf{r}_n,i_n) \in (\Delta\times\mathbb{N})^+$*, where* r_1 *is the rule* $\mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \leftarrow \phi \in \Delta$*, the* unfolding $\mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \xrightarrow{(\mathsf{r}_1,i_1)\ldots(\mathsf{r}_n,i_n)}_{\Delta} \psi$ *is inductively defined as (1)* $\psi = \phi$ *if* $n = 1$*, and (2)* ψ *is obtained from* ϕ *by replacing its* i_1*-th predicate atom* $\mathsf{B}(y_1,\ldots,y_{\#\mathsf{B}})$ *with* $\psi_1[x_1/y_1,\ldots,x_{\#\mathsf{B}}/y_{\#\mathsf{B}}]$*, where* $\mathsf{B}(x_1,\ldots,x_{\#\mathsf{B}}) \xrightarrow{(\mathsf{r}_2,i_2)\ldots(\mathsf{r}_n,i_n)}_{\Delta} \psi_1$ *is an unfolding, if* $n > 1$.

We show that the $\mathsf{Bnd}[\Delta,\mathsf{A}]$ problem can be reduced to the existence of increasingly large unfoldings or, equivalently, a cycle in a finite directed graph, built by a variant of the least fixpoint iteration algorithm used to solve the satisfiability problem (Fig. 3).

Definition 10. *Given satisfiable base pairs* $\mathsf{t},\mathsf{u} \in \mathsf{SatBase}$ *and a rule from* Δ*:*

$$\mathsf{r} \;:\; \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \mathsf{B}_1(z_1^1,\ldots,z_{\#\mathsf{B}_1}^1) * \ldots * \mathsf{B}_h(z_1^h,\ldots,z_{\#\mathsf{B}_h}^h)$$

where ϕ *is a quantifier- and predicate-free formula, we write* $(\mathsf{A},\mathsf{t}) \overset{(\mathsf{r},i)}{\rightsquigarrow\!\triangleright} (\mathsf{B},\mathsf{u})$ *if and only if* $\mathsf{B} = \mathsf{B}_i$ *and there exist satisfiable base tuples* $\mathsf{t}_1,\ldots,\mathsf{u} = \mathsf{t}_i,\ldots,\mathsf{t}_h \in \mathsf{SatBase}$*, such that* $\mathsf{t} \in \big(\mathsf{Base}(\phi,\{x_1,\ldots,x_{\#\mathsf{A}}\}) \otimes \bigotimes_{\ell=1}^{h} \mathsf{t}_\ell[x_1/z_1^\ell,\ldots,x_{\#\mathsf{B}_\ell}/z_{\#\mathsf{B}_\ell}^\ell]\big)\downarrow_{x_1,\ldots,x_{\#\mathsf{A}}}$*. We define the directed graph with edges labeled by pairs* $(\mathsf{r},i) \in \Delta\times\mathbb{N}$*:*

$$\mathcal{G}(\Delta) \overset{\mathsf{def}}{=} \big(\{\mathsf{def}(\Delta)\times\mathsf{SatBase}\}, \{\langle(\mathsf{A},\mathsf{t}),(\mathsf{r},i),(\mathsf{B},\mathsf{u})\rangle \mid (\mathsf{A},\mathsf{t}) \overset{(\mathsf{r},i)}{\rightsquigarrow\!\triangleright} (\mathsf{B},\mathsf{u})\}\big)$$

input: a SID Δ
output: $\mathcal{G}(\Delta) = (V, E)$

initially $V := \emptyset$, $E := \emptyset$
for $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi \in \Delta$, with ϕ quantifier- and predicate-free **do**
    $V := V \cup \left(\{\mathsf{A}\} \times \mathsf{Base}(\phi, \{x_1, \ldots, x_{\#\mathsf{A}}\})\downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}\right)$
while V or E still change **do**
    for $\mathsf{r} : \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \divideontimes_{\ell=1}^{h} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell) \in \Delta$ **do**
        if there exist $(\mathsf{B}_1, \mathsf{t}_1), \ldots, (\mathsf{B}_h, \mathsf{t}_h) \in V$ **then**
            $\mathcal{X} := \left(\mathsf{Base}(\phi, \{x_1, \ldots, x_{\#\mathsf{A}}\}) \otimes \bigotimes_{\ell=1}^{h} \mathsf{t}_\ell[x_1/z_1^\ell, \ldots, x_{\#\mathsf{B}_\ell}/z_{\#\mathsf{B}_\ell}^\ell]\right)\downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}$
            $V := V \cup (\{\mathsf{A}\} \times \mathcal{X})$
            $E := E \cup \{\langle (\mathsf{A}, \mathsf{t}), (\mathsf{r}, \ell), (\mathsf{B}_\ell, \mathsf{t}_\ell)\rangle \mid \mathsf{t} \in \mathcal{X}, \ell \in [1, h]\}$

Fig. 3: Algorithm for the Construction of $\mathcal{G}(\Delta)$

The graph $\mathcal{G}(\Delta)$ is built by the algorithm in Fig. 3, a slight variation of the classical Kleene iteration algorithm for the computation of the least solution of the constraints of the form (1)(see Fig. 2). A path $(\mathsf{A}_1, \mathsf{t}_1) \overset{(\mathsf{r}_1, i_1)}{\rightsquigarrow} (\mathsf{A}_2, \mathsf{t}_2) \overset{(\mathsf{r}_2, i_2)}{\rightsquigarrow} \ldots \overset{(\mathsf{r}_n, i_n)}{\rightsquigarrow} (\mathsf{A}_n, \mathsf{t}_n)$ in $\mathcal{G}(\Delta)$ induces a unique unfolding $\mathsf{A}_1(x_1, \ldots, x_{\#\mathsf{A}_1}) \overset{(\mathsf{r}_1, i_1) \ldots (\mathsf{r}_n, i_n)}{\Longrightarrow}_\Delta \phi$ (Def. 9). Since the vertices of $\mathcal{G}(\Delta)$ are pairs (A, t), where t is a satisfiable base tuple and the edges of $\mathcal{G}(\Delta)$ reflect the construction of the base tuples from the least solution of the constraints (1), the outcome ϕ of this unfolding is always a satisfiable formula.

Lemma 8. *Given a path* $(\mathsf{A}_0, \mathsf{t}_0) \overset{(\mathsf{r}_1, i_1)}{\rightsquigarrow} \ldots \overset{(\mathsf{r}_n, i_n)}{\rightsquigarrow} (\mathsf{A}_n, \mathsf{t}_n)$ *in* $\mathcal{G}(\Delta)$*, where* $\mathsf{t}_0 = (\mathcal{C}^\sharp, I^\sharp, \pi)$*, a state map* ρ *and a store* ν*, such that* $(\emptyset, \emptyset, \rho) \models^\nu \pi$*, there exists a configuration* $(\mathcal{C}, I, \rho)$*, such that* $(\mathcal{C}, I, \rho) \models^\nu_\Delta \phi$*, where* $\mathsf{A}_0(x_1, \ldots, x_{\#\mathsf{A}_0}) \overset{(\mathsf{r}_1, i_1) \ldots (\mathsf{r}_n, i_n)}{\Longrightarrow}_\Delta \phi$ *is the unique unfolding corresponding to the path.*

Proof. Let r_1 be the following rule:

$$\mathsf{A}_0(x_1, \ldots, x_{\#\mathsf{A}_0}) \leftarrow \phi, \text{ where } \phi = \exists y_1 \ldots \exists y_m \,.\, \psi * \pi * \divideontimes_{\ell=2}^{h} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$$

and $\psi * \pi$ is a quantifier- and predicate-free formula and π is, moreover, pure. The proof goes by induction on the length $n \geq 1$ of the path. For the base case $n = 1$, by Def. 10, the edge $(\mathsf{A}_0, \mathsf{t}_0) \overset{(\mathsf{r}_1, i_1)}{\rightsquigarrow} (\mathsf{A}_1, \mathsf{t}_1)$ implies the existence of base tuples $\mathsf{u}_\ell \in \overrightarrow{\mu X}.\Delta^\sharp(\mathsf{B}_\ell)$, for all $\ell \in [2, h]$, such that $\mathsf{B}_{i_1} = \mathsf{A}_1$, $\mathsf{u}_{i_1 - 1} = \mathsf{t}_1$ and:

$$\mathsf{t}_0 \in \left(\mathsf{Base}(\psi * \pi, \{x_1, \ldots, x_{\#\mathsf{A}}\}) \otimes \bigotimes_{\ell=2}^{h} \mathsf{u}_\ell[x_1/z_1^\ell, \ldots, x_{\#\mathsf{B}_\ell}/z_{\#\mathsf{B}_\ell}^\ell]\right)\downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}$$

Let $\mathsf{u}_\ell \overset{\text{def}}{=} (\mathcal{C}_\ell^\sharp, I_\ell^\sharp, \pi_\ell)$, for all $\ell \in [2, h]$ and $\pi' \overset{\text{def}}{=} \pi * \divideontimes_{\ell=2}^{h} \pi_\ell$. Since t_0 is satisfiable, there exists a store ν', that agrees with ν over $x_1, \ldots, x_{\#\mathsf{A}_0}$, such that, moreover:

$$\nu'(x) = \nu'(y) \text{ only if } x \approx_{\pi'} y, \text{ for all } x, y \in \mathrm{fv}(\psi * \pi') \cup \bigcup_{\ell=2}^{h} \left(\mathcal{C}_\ell^\sharp \cup \{z_i \mid \langle z_1, \ldots, z_n\rangle \in I_\ell^\sharp(\tau), \tau \in \mathsf{Inter}\}\right) \qquad (\dagger)$$

We define the configurations $(\mathcal{C}_1, I_1, \rho), \ldots, (\mathcal{C}_h, I_h, \rho)$ inductively, as follows:

- $\mathcal{C}_1 \stackrel{\text{def}}{=} \{\nu'(y) \mid [y] \text{ occurs in } \psi\}$,
- $I_1 \stackrel{\text{def}}{=} \{(\nu'(z_1), p_1, \ldots, \nu'(z_s), p_s) \mid \langle z_1.p_1, \ldots, z_t.p_t\rangle \text{ occurs in } \psi\}$,
- for all $\ell \in [2,h]$, assuming $\mathcal{C}_1, \ldots, \mathcal{C}_{\ell-1}$ and $I_1, \ldots, I_{\ell-1}$ are defined, let:

$$\mathcal{D}_\ell \stackrel{\text{def}}{=} \bigcup_{i=1}^{\ell-1} \mathcal{C}_i \cup \bigcup_{i=\ell+1}^{h} \nu'(\mathcal{C}_i^\sharp)$$
$$\mathcal{J}_\ell \stackrel{\text{def}}{=} \bigcup_{i=1}^{\ell-1} I_i \cup \bigcup_{i=\ell+1}^{h} \nu'(I_i^\sharp)$$

We prove first that $\mathcal{D}_\ell \cap \nu'(\mathcal{C}_\ell^\sharp) = \emptyset$ and $\mathcal{J}_\ell \cap \nu'(I_\ell^\sharp) = \emptyset$ (we prove only the first point, the second uses a similar reasoning), by induction on $\ell \in [2,h]$. For the base case $\mathcal{D}_2 \cap \nu'(\mathcal{C}_2^\sharp) \neq \emptyset$, we prove the points below:

- $\mathcal{C}_1 \cap \nu'(\mathcal{C}_2^\sharp) = \emptyset$: suppose, for a contradiction, that there exists $c \in \mathcal{C}_1 \cap \nu'(\mathcal{C}_2^\sharp)$, then $c = \nu'(y)$, for a component atom $[y]$ from ψ and $c = \nu'(x)$, for some $x \in \mathcal{C}_2^\sharp$. By (†), we obtain $x \approx_{\pi'} y$, contradicting the existence of $\mathfrak{t}_0$.
- $\nu'(\mathcal{C}_i^\sharp) \cap \nu'(\mathcal{C}_2^\sharp) = \emptyset$, for some $i \in [3,h]$: suppose, for a contradiction, that there exists $c \in \nu'(\mathcal{C}_i^\sharp) \cap \nu'(\mathcal{C}_2^\sharp)$, then $c = \nu'(x) = \nu'(y)$, for some $x \in \mathcal{C}_i^\sharp$ and $y \in \mathcal{C}_2^\sharp$. By (†), we obtain $x \approx_{\pi'} y$, contradicting the existence of $\mathfrak{t}_0$.

We assume that $\mathcal{D}_j \cap \nu'(\mathcal{C}_j^\sharp) = \emptyset$, for all $j \in [2, \ell-1]$. By Lemma 3, there exist configurations $(\mathcal{C}_j, I_j, \rho)$, such that $\mathcal{C}_j \cap \mathcal{D}_j = \emptyset$ (‡) and $(\mathcal{C}_j, I_j, \rho) \models^{\nu'} \mathsf{B}_j(x_1, \ldots, x_{\#\mathsf{B}_j})$, for all $j \in [2, \ell-1]$. We prove $\mathcal{D}_\ell \cap \nu'(\mathcal{C}_\ell^\sharp) = \emptyset$, by showing the following points:

- $\mathcal{C}_j \cap \nu'(\mathcal{C}_\ell^\sharp) = \emptyset$, for all $j \in [1, \ell-1]$: suppose, for a contradiction, that there exists $c \in \mathcal{C}_j \cap \nu'(\mathcal{C}_\ell^\sharp)$, for some $j \in [1, \ell-1]$, then $c \in \mathcal{C}_j \cap \mathcal{D}_j$, because $\mathcal{D}_j \subseteq \nu'(\mathcal{C}_\ell^\sharp)$, in contradiction with $\mathcal{C}_j \cap \mathcal{D}_j = \emptyset$ (‡).
- $\nu'(\mathcal{C}_j^\sharp) \cap \nu'(\mathcal{C}_\ell^\sharp) = \emptyset$, for all $j \in [2, \ell-1]$: suppose, for a contradiction, that there exists $c \in \nu'(\mathcal{C}_j^\sharp) \cap \nu'(\mathcal{C}_\ell^\sharp)$, then $c = \nu'(x) = \nu'(y)$, for some $x \in \mathcal{C}_j^\sharp$ and $y \in \mathcal{C}_\ell^\sharp$. By (†), we obtain $x \approx_{\pi'} y$, contradicting the existence of $\mathfrak{t}_0$.

Consequently, $\mathcal{D}_\ell \cap \nu'(\mathcal{C}_\ell^\sharp) = \emptyset$, for all $\ell \in [2,h]$. By Lemma 3, there exists a configuration $(\mathcal{C}_\ell, I_\ell, \rho)$, such that $\mathcal{C}_\ell \cap \mathcal{D}_\ell = \emptyset$ and $(\mathcal{C}_\ell, I_\ell, \rho) \models^{\nu'} \mathsf{B}_\ell(x_1, \ldots, x_{\#\mathsf{B}_\ell})$, for all $\ell \in [2,h]$. We obtain that $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$ and $I_i \cap I_j = \emptyset$, for all $1 \leq i < j \leq h$, meaning that the configuration $(\mathcal{C}, I, \rho) \stackrel{\text{def}}{=} (\mathcal{C}_1, I_1, \rho) \bullet \ldots \bullet (\mathcal{C}_h, I_h, \rho)$ is defined, which leads to $(\mathcal{C}, I, \rho) \models^{\nu'} \phi$.

For the inductive step $n > 1$, by Def. 10, there exists base tuples $\mathfrak{u}_\ell \in \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{B}_\ell)$, for all $\ell \in [2,h]$, such that $\mathsf{A}_1 = \mathsf{B}_{i_1}$, $\mathfrak{u}_{i_1-1} = \mathfrak{t}_1$ and:

$$\mathfrak{t}_0 \in \left(\mathsf{Base}(\psi * \pi, \{x_1, \ldots, x_{\#\mathsf{A}}\}) \otimes \bigotimes_{\ell=1}^{h} \mathfrak{u}_\ell[x_1/z_1^\ell, \ldots, x_{\#\mathsf{B}_\ell}/z_{\#\mathsf{B}_\ell}^\ell] \right) \downarrow_{x_1, \ldots, x_{\#\mathsf{A}}}$$

Then there exists a store ν' that agrees with ν over $x_1, \ldots, x_{\#\mathsf{A}_0}$ and satisfies (†). Let $\nu'' \stackrel{\text{def}}{=} \nu'[x_1/z_1^{i_1}, \ldots, x_{\#\mathsf{A}_1}/z_{\#\mathsf{A}_1}^{i_1}]$. By the inductive hypothesis, since $(\mathsf{A}_1, \mathfrak{t}_1) \overset{(r_2,i_2)}{\rightsquigarrow} \ldots \overset{(r_n,i_n)}{\rightsquigarrow} (\mathsf{A}_n, \mathfrak{t}_n)$ is a path in $\mathcal{G}(\Delta)$, there exists a configuration $(\mathcal{C}', I', \rho)$, such that $(\mathcal{C}', I', \rho) \models^{\nu''} \mathsf{A}_1(z_1^{i_1}, \ldots, z_{\#\mathsf{A}_1}^{i_1})$, because $(\mathcal{C}', I', \rho) \models^{\nu'} \phi_1$, for the unfolding $\mathsf{A}_1(x_1, \ldots, x_{\#\mathsf{A}_1}) \xrightarrow{(r_2,i_2)\ldots(r_n,i_n)}_{\Delta} \phi_1$. The required configuration is defined as $(\mathcal{C}, I, \rho) \stackrel{\text{def}}{=} (\mathcal{C}_1, I_1, \rho) \bullet \ldots \bullet (\mathcal{C}_{i_1-1}, I_{i_1-1}, \rho) \bullet$

$(\mathcal{C}', I', \rho) \bullet (\mathcal{C}_{i_1+1}, I_{i_1+1}, \rho) \bullet \ldots \bullet (\mathcal{C}_h, I_h, \rho)$, where $(\mathcal{C}_1, I_1, \rho)$, ..., $(\mathcal{C}_{i_1-1}, I_{i_1-1}, \rho)$ and $(\mathcal{C}_{i_1+1}, I_{i_1+1}, \rho)$, ..., $(\mathcal{C}_h, I_h, \rho)$ are defined as in the base case, by taking ν'' instead of ν' and defining, for all $\ell \in [2,h] \setminus \{i_1\}$:

$$\mathcal{D}_\ell \stackrel{\text{def}}{=} \mathcal{C}' \cup \bigcup_{i=1}^{\ell-1} \mathcal{C}_i \cup \bigcup_{i=\ell+1}^{h} \nu'(\mathcal{C}_i^\sharp)$$
$$\mathcal{I}_\ell \stackrel{\text{def}}{=} I' \cup \bigcup_{i=1}^{\ell-1} I_i \cup \bigcup_{i=\ell+1}^{h} \nu'(I_i^\sharp)$$

The proof of the fact that $\mathcal{C}_1$, ..., $\mathcal{C}_{i_1-1}$, $\mathcal{C}'$, $\mathcal{C}_{i_1+1}$, ..., $\mathcal{C}_h$ and I_1, ..., I_{i_1-1}, I', I_{i_1+1}, ..., I_h are pairwise disjoint, respectively, follows by the same argument as in the base case. □

Lemma 9. *Given an unfolding* $\mathsf{A}_0(x_1, \ldots, x_{\#\mathsf{A}_0}) \xRightarrow{(\mathsf{r}_1,i_1)\ldots(\mathsf{r}_n,i_n)}_\Delta \phi$, *a configuration* $(\mathcal{C}, I, \rho)$ *and a store* ν, *such that* $(\mathcal{C}, I, \rho) \models^\nu_\Delta \phi$, *then* $\mathcal{G}(\Delta)$ *has a path* $(\mathsf{A}_0, (\mathcal{C}_0^\sharp, I_0^\sharp, \pi_0)) \overset{(\mathsf{r}_1,i_1)}{\rightsquigarrow} \ldots \overset{(\mathsf{r}_n,i_n)}{\rightsquigarrow} (\mathsf{A}_n, (\mathcal{C}_n^\sharp, I_n^\sharp, \pi_n))$, *for some* $(\mathcal{C}_0^\sharp, I_0^\sharp, \pi_0), \ldots,$
$(\mathcal{C}_n^\sharp, I_n^\sharp, \pi_n) \in \mathsf{SatBase}$, *such that* $\nu(\mathcal{C}_0^\sharp) \subseteq \mathcal{C}_0$, $\nu(I_0^\sharp) \subseteq I_0$ *and* $(\emptyset, \emptyset, \rho) \models^\nu \pi_0$.

Proof. Let r_1 be the following rule:

$$\mathsf{A}_0(x_1, \ldots, x_{\#\mathsf{A}_0}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \psi * \pi * \circledast_{\ell=2}^{h} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$$

and $\psi * \pi$ is a quantifier- and predicate-free formula and π is, moreover, pure. The proof goes by induction on the length $n \geq 1$ of the path. For the base case $n = 1$, we have $\phi = \exists y_1 \ldots \exists y_m \,.\, \psi * \pi * \circledast_{\ell=2}^{h} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$, hence there exists a store ν', that agrees with ν over $x_1, \ldots, x_{\#\mathsf{A}_0}$, and configurations $(\mathcal{C}_1, I_1, \rho), \ldots, (\mathcal{C}_h, I_h, \rho)$, such that:

- $(\mathcal{C}_1, I_1, \rho) \models^{\nu'} \psi * \pi$,
- $(\mathcal{C}_\ell, I_\ell, \rho) \models^{\nu'}_\Delta \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$, for all $\ell \in [2,h]$, and
- $\gamma = (\mathcal{C}_1, I_1, \rho) \bullet \ldots \bullet (\mathcal{C}_h, I_h, \rho)$.

We consider the following base tuples:

- $(\overline{\mathcal{C}}_1^\sharp, \overline{I}_1^\sharp, \overline{\pi}_1) \stackrel{\text{def}}{=} \mathsf{Base}(\psi * \pi, \{x_1, \ldots, x_{\#\mathsf{A}_0}\})$,
- for all $\ell \in [2,h]$, there exist $(\overline{\mathcal{C}}_\ell^\sharp, \overline{I}_\ell^\sharp, \overline{\pi}_\ell) \in \mu\overrightarrow{X}.\Delta^\sharp(\mathsf{B}_\ell)[x_1/z_1^\ell, \ldots, x_{\#\mathsf{B}_\ell}/z_{\#\mathsf{B}_\ell}^\ell]$, such that $\mathcal{C}_\ell \subseteq \nu'(\overline{\mathcal{C}}_\ell^\sharp)$, $I_\ell \subseteq \nu'(\overline{I}_\ell^\sharp)$ and $(\emptyset, \emptyset, \rho) \models^{\nu'} \overline{\pi}_\ell$, by Lemma 4.

By similar argument to the one from the proof of Lemma 4 (base case), we show that the composition $\bigotimes_{\ell=1}^{h}(\overline{\mathcal{C}}_\ell^\sharp, \overline{I}_\ell^\sharp, \overline{\pi}_\ell)$ is defined and let $(\mathcal{C}_0^\sharp, I_0^\sharp, \pi_0) \stackrel{\text{def}}{=} \left(\bigotimes_{\ell=1}^{h}(\overline{\mathcal{C}}_\ell^\sharp, \overline{I}_\ell^\sharp, \overline{\pi}_\ell)\right)\downarrow_{x_1,\ldots,x_{\#\mathsf{A}}}$. Moreover, we obtain $\nu(\mathcal{C}_0^\sharp) \subseteq \mathcal{C}_0$, $\nu(I_0^\sharp) \subseteq I_0$ and $(\emptyset, \emptyset, \rho) \models^\nu \pi_0$, as in the proof of Lemma 4. Then, by Def. 10, $\mathcal{G}(\Delta)$ has an edge $(\mathsf{A}_0, (\mathcal{C}_0^\sharp, I_0^\sharp, \pi_0)) \overset{(\mathsf{r}_1,i_1)}{\rightsquigarrow} (\mathsf{B}_{i_1-1}, (\overline{\mathcal{C}}_{i_1-1}^\sharp, \overline{I}_{i_1-1}^\sharp, \overline{\pi}_{i_1-1}))$.

For the inductive step $n > 1$, let $\mathsf{B}_{i_1-1}(x_1, \ldots, x_{\#\mathsf{B}_{i_1-1}}) \xRightarrow{(\mathsf{r}_2,i_2)\ldots(\mathsf{r}_n,i_n)}_\Delta \phi_1$ be an unfolding, such that ϕ is obtained from $\exists y_1 \ldots \exists y_m \,.\, \psi * \pi * \circledast_{\ell=2}^{h} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$, by replacing $\mathsf{B}_{i_1-1}(z_1^{i_1-1}, \ldots, z_{\#\mathsf{B}_{i_1-1}}^{i_1-1})$ with $\phi_1[x_1/z_1^{i_1-1}, \ldots, x_{\#\mathsf{B}_{i_1-1}}/z_{\#\mathsf{B}_{i_1-1}}^{i_1-1}]$. Since $(\mathcal{C}, I, \rho) \models^\nu_\Delta \phi$, there exists a store ν', that agrees with ν over $x_1, \ldots, x_{\#\mathsf{A}_0}$, and configurations $(\mathcal{C}_1, I_1, \rho), \ldots, (\mathcal{C}_h, I_h, \rho)$, where $\gamma = (\mathcal{C}_1, I_1, \rho) \bullet \ldots \bullet (\mathcal{C}_h, I_h, \rho)$ and the following hold:

- $(\mathcal{C}_1, I_1, \rho) \models^{\nu'}_{\Delta} \psi * \pi$,
- $(\mathcal{C}_\ell, I_\ell, \rho) \models^{\nu'}_{\Delta} \mathsf{B}_\ell(z^\ell_1, \ldots, z^\ell_{\#\mathsf{B}_\ell})$, for all $\ell \in [2,h] \setminus \{i_1+1\}$,
- $(\mathcal{C}_{i_1-1}, I_{i_1-1}, \rho) \models^{\nu'}_{\Delta} \phi_1[x_1/z^{i_1-1}_1, \ldots, x_{\#\mathsf{B}_{i_1-1}}/z^{i_1-1}_{\#\mathsf{B}_{i_1-1}}]$, hence $(\mathcal{C}_{i_1-1}, I_{i_1-1}, \rho) \models^{\nu'}_{\Delta} \mathsf{B}_{i_1-1}(z^{i_1-1}_1, \ldots, z^{i_1-1}_{\#\mathsf{B}_{i_1-1}})$.

We consider the following base tuples:

- $(\overline{\mathcal{C}}^\sharp_1, \overline{I}^\sharp_1, \overline{\pi}_1) \stackrel{\text{def}}{=} \mathsf{Base}(\psi * \pi, \{x_1, \ldots, x_{\#\mathsf{A}_0}\})$,
- for all $\ell \in [2,h] \setminus \{i_1+1\}$, there exist $(\overline{\mathcal{C}}^\sharp_\ell, \overline{I}^\sharp_\ell, \overline{\pi}_\ell) \in \mu\overrightarrow{\mathcal{X}}.\Delta^\sharp(\mathsf{B}_\ell)[x_1/z^\ell_1, \ldots, x_{\#\mathsf{B}_\ell}/z^\ell_{\#\mathsf{B}_\ell}]$, such that $\mathcal{C}_\ell \subseteq \nu'(\overline{\mathcal{C}}^\sharp_\ell)$, $I_\ell \subseteq \nu'(\overline{I}^\sharp_\ell)$ and $(\emptyset, \emptyset, \rho) \models^{\nu'} \overline{\pi}_\ell$, by Lemma 4.
- $\mathcal{G}(\Delta)$ has a path $(\mathsf{B}_{i_1-1}, (\mathcal{C}^\sharp_1, I^\sharp_1, \pi_1)) \overset{(r_2,i_2)}{\rightsquigarrow} \ldots \overset{(r_n,i_n)}{\rightsquigarrow} (\mathsf{A}_n, (\mathcal{C}^\sharp_n, I^\sharp_n, \pi_n))$, such that $\mathcal{C}_{i_1-1} \subseteq \nu'(\mathcal{C}^\sharp_1)$, $I_{i_1-1} \subseteq \nu'(I^\sharp_1)$ and $(\emptyset, \emptyset, \rho) \models^{\nu'} \pi_1$, by the inductive hypothesis.

By an argument similar to the one from Lemma 4, the composition $(\overline{\mathcal{C}}^\sharp, \overline{I}^\sharp, \overline{\pi}) \stackrel{\text{def}}{=} \bigotimes_{\ell=1}^{i_1-2} (\overline{\mathcal{C}}^\sharp_\ell, \overline{I}^\sharp_\ell, \overline{\pi}_\ell) \otimes (\mathcal{C}^\sharp_1, I^\sharp_1, \pi_1) \otimes \bigotimes_{\ell=i_1}^{h} (\overline{\mathcal{C}}^\sharp_\ell, \overline{I}^\sharp_\ell, \overline{\pi}_\ell)$ is defined and let $(\mathcal{C}^\sharp_0, I^\sharp_0, \pi_0) \stackrel{\text{def}}{=} (\overline{\mathcal{C}}^\sharp, \overline{I}^\sharp, \overline{\pi}) \downarrow_{\{x_1, \ldots, x_{\#\mathsf{A}_0}\}}$. Finally, the conditions $\nu(\mathcal{C}^\sharp_0) \subseteq \mathcal{C}_0$, $\nu(I^\sharp_0) \subseteq I_0$ and $(\emptyset, \emptyset, \rho) \models^{\nu} \pi_0$ follow from a similar argument to the one used in Lemma 4. □

An *elementary cycle* of $\mathcal{G}(\Delta)$ is a path from some vertex $(\mathsf{B}, \mathfrak{u})$ back to itself, such that $(\mathsf{B}, \mathfrak{u})$ does not occur on the path, except at its endpoints. The cycle is, moreover, *reachable* from $(\mathsf{A}, \mathfrak{t})$ if and only if there exists a path $(\mathsf{A}, \mathfrak{t}) \overset{(r_1,i_1)}{\rightsquigarrow} \ldots \overset{(r_n,i_n)}{\rightsquigarrow} (\mathsf{B}, \mathfrak{u})$ in $\mathcal{G}(\Delta)$. We reduce the complement of the $\mathsf{Bnd}[\Delta, \mathsf{A}]$ problem, namely the existence of an infinite set of models of $\exists x_1 \ldots \exists x_{\#\mathsf{A}} \,.\, \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$ of unbounded degree, to the existence of a reachable elementary cycle in $\mathcal{G}(\Delta')$, where Δ' is obtained from Δ, as described in the following.

First, we consider, for each predicate $\mathsf{B} \in \mathrm{def}(\Delta)$, a predicate B', of arity $\#\mathsf{B}+1$, not in $\mathrm{def}(\Delta)$ i.e., the set of predicates for which there exists a rule in Δ. Second, for each rule $\mathsf{B}_0(x_1, \ldots, x_{\#\mathsf{B}_0}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \circledast_{\ell=2}^{h} \mathsf{B}_\ell(z^\ell_1, \ldots, z^\ell_{\#\mathsf{B}_\ell}) \in \Delta$, where ϕ is a quantifier- and predicate-free formula and $\mathrm{iv}(\phi) \subseteq \mathrm{fv}(\phi)$ denotes the subset of variables occurring in interaction atoms in ϕ, the SID Δ' has the following rules:

$$\mathsf{B}'_0(x_1, \ldots, x_{\#\mathsf{B}_0}, x_{\#\mathsf{B}_0+1}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \circledast_{\xi \in \mathrm{iv}(\phi)} x_{\#\mathsf{B}_0+1} \neq \xi * \circledast_{\ell=2}^{h} \mathsf{B}'_\ell(z^\ell_1, \ldots, z^\ell_{\#\mathsf{B}_\ell}, x_{\#\mathsf{B}_0+1}) \tag{2}$$

$$\mathsf{B}'_0(x_1, \ldots, x_{\#\mathsf{B}_0}, x_{\#\mathsf{B}_0+1}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * x_{\#\mathsf{B}_0+1} = \xi * \circledast_{\ell=2}^{h} \mathsf{B}'_\ell(z^\ell_1, \ldots, z^\ell_{\#\mathsf{B}_\ell}, x_{\#\mathsf{B}_0+1}) \tag{3}$$

for each variable $\xi \in \mathrm{iv}(\phi)$, that occurs in an interaction atom in ϕ.

Intuitively, there exists a family of models (with respect to Δ) of $\exists x_1 \ldots \exists x_{\#\mathsf{A}} \,.\, \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$ of unbounded degree if and only if these are models of $\exists x_1 \ldots \exists x_{\#\mathsf{A}+1} \,.\, \mathsf{A}'(x_1, \ldots, x_{\#\mathsf{A}+1})$ (with respect to Δ') and the last parameter of each predicate $\mathsf{B}' \in \mathrm{def}(\Delta')$ can be mapped, in each of the these models, to a component that occurs in unboundedly many interactions. The latter condition is equivalent to the existence of an elementary cycle, containing a rule of the form (3), that it, moreover, reachable from some vertex $(\mathsf{A}', \mathfrak{t})$ of $\mathcal{G}(\Delta')$, for some $\mathfrak{t} \in \mathsf{SatBase}$. This reduction is formalized below:

Lemma 10. *Let* A *be a predicate and* γ *be a model of* $\exists x_1 \ldots \exists x_{\#\mathsf{A}} \,.\, \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$. *Then there exists an unfolding* $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \overset{w}{\Rightarrow}_{\Delta} \psi$ *of length* $|w| \geq \frac{\log(\delta(\gamma)) - \log \beta_1}{\log \beta_2}$ *where* β_1

is the maximal number of components and interaction atoms and β_2 *is the maximal number of predicate atoms, occurring in a rule of* Δ.

Proof. Let γ be a configuration, ν be a store and A be a predicate, such that $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}})$. We consider the derivation tree T induced by the definition of the $\models^{\nu}_{\Delta}$ relation. The nodes of T are labelled by a triple $\gamma' \models^{\nu'}_{\Delta} \mathsf{B}(x_1,\ldots,x_{\#\mathsf{B}})$. We start from the root labelled by $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}})$ and define the children of a node inductively.

For each node $\gamma' \models^{\nu'}_{\Delta} \mathsf{B}(x_1,\ldots,x_{\#\mathsf{B}})$, there exists a rule:

$$\mathsf{r} \;:\; \mathsf{B}(x_1,\ldots,x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \mathsf{B}_1(z^1_1,\ldots,z^1_{\#\mathsf{B}_1}) * \ldots * \mathsf{B}_h(z^h_1,\ldots,z^h_{\#\mathsf{B}_h})$$

and configurations $\gamma_0,\ldots,\gamma_h$, such that $\gamma = \gamma_0 \bullet \ldots \bullet \gamma_h$, $\gamma_0 \models^{\nu''}_{\Delta} \phi$ and $\gamma_i \models^{\nu''}_{\Delta} \mathsf{B}_i(z^i_1,\ldots,z^i_{\#\mathsf{B}_i})$ for every $i \in [1,h]$, where ϕ is a predicate-free formula and ν'' is a store that agrees with ν' over $x_1,\ldots,x_{\#\mathsf{B}}$. We define $\nu_\ell \stackrel{\text{def}}{=} \nu''[x_1/z^\ell,\ldots,x_{\#\mathsf{B}_\ell}/z^\ell_{\#\mathsf{B}_\ell}]$, for all $\ell \in [1,h]$. Then the node $\gamma' \models^{\nu'}_{\Delta} \mathsf{B}(x_1,\ldots,x_{\#\mathsf{B}})$ has h children in T, where the ℓ-th child is labelled by $\gamma_\ell \models^{\nu_\ell}_{\Delta} \mathsf{B}_\ell(x_1,\ldots,x_{\#\mathsf{B}_\ell})$, for all $\ell \in [1,h]$. The construction is finite since $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}})$ has a finite inductive definition.

We now consider the degree of the configurations which occur in T. By Def. 3, we obtain $\delta(\gamma) \leq \delta(\gamma_0) + \sum_{i=1}^{h} \delta(\gamma_i)$. With each γ_i associated to a child of this node (except γ_0), we obtain that $\delta(\gamma_{\text{init}})$ does not exceed β_1 times the number of nodes in T. Since $h \leq \beta_2$, the height n of T is bound to the degree $\delta(\gamma)$ by the inequality $\delta(\gamma) \leq \beta_1 \times \sum_{k=0}^{n} {\beta_2}^k = \beta_1 \times {\beta_2}^{n+1} - 1$, leading to:

$$n+1 \geq \frac{\log \delta(\gamma) - \log \beta_1}{\log \beta_2}$$

Finally, with T of height n, there exists a branch in T (starting from the root) of length exactly $n+1$. Yet each branch of T corresponds to an unfolding $\mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \stackrel{w}{\Rightarrow}_{\Delta} \psi$, with w obtained by concatenating for every node (γ,ν,A) of the branch (from root to leaf) the couple (r,i) consisting of:

- the rule $\mathsf{r} \in \Delta$ used to unfold $\mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}})$, and
- the position i of this node among its brothers in T (take $i = 1$ for the root).

This unfolding has the length required, which concludes the lemma. □

Lemma 11. *There exists an infinite sequence of configurations* $\gamma_1,\gamma_2,\ldots$ *such that* $\gamma_i \models_{\Delta} \exists x_1 \ldots \exists x_{\#\mathsf{A}} \,.\, \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}})$ *and* $\delta(\gamma_i) < \delta(\gamma_{i+1})$*, for all* $i \geq 1$ *if and only if* $\mathcal{G}(\Delta')$ *has an elementary cycle containing a rule (3), reachable from a node* (A',t)*, for* $\mathsf{t} \in \mathsf{SatBase}$.

Proof. "⇒" Let $\nu_1,\nu_2,\ldots$ be stores such that $\gamma_i \models^{\nu_i}_{\Delta} \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}})$, for all $i \geq 1$. By Lemma 10, where exists unfoldings $\mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \stackrel{w_i}{\Longrightarrow}_{\Delta'} \phi_i$ of lengths $|w_1| < |w_2| < \ldots$, such that $\gamma_i \models^{\nu_i}_{\Delta} \phi_i$, for all $i \geq 1$. For each configuration γ_i, let $d_i \in \mathbb{C}$ be a component, such that $\delta(\gamma_i) = \|\{(c_1,p_1,\ldots,c_n,p_n) \mid d_i = c_j, j \in [1,n]\}\|$. By induction on $|w_i| \geq 1$, we build unfoldings $\mathsf{A}'(x_1,\ldots,x_{\#\mathsf{A}},x_{\#\mathsf{A}+1}) \stackrel{w'_i}{\Longrightarrow}_{\Delta'} \phi'_i$ that bind $x_{\#\mathsf{A}+1}$ to all variables bound to d_i, using rules of type (3). By Lemma 9, $w'_1,w'_2,\ldots$ are labels of paths from $\mathcal{G}(\Delta')$, that

start in $(\mathsf{A},\mathsf{t}_1),(\mathsf{A},\mathsf{t}_2),\ldots$, respectively. Since $\mathcal{G}(\Delta')$ is finite, we can chose an infinite subsequence of paths that start in the same node of $\mathcal{G}(\Delta')$ and repeat the same vertex, with a rule of type (3) in between.

"$\Leftarrow$" Let $(\mathsf{A}',\mathsf{t}) \overset{(\mathsf{r}'_1,i_1)}{\rightsquigarrow} \ldots \overset{(\mathsf{r}'_n,i_n)}{\rightsquigarrow} (\mathsf{B}_n,\mathsf{t}_n) \overset{(\mathsf{r}'_{n+1},i_{n+1})}{\rightsquigarrow} \ldots \overset{(\mathsf{r}'_{n+p},i_{n+p})}{\rightsquigarrow} (\mathsf{B}_n,\mathsf{t}_n)$ be a path in $\mathcal{G}(\Delta')$, such that one of the rules $\mathsf{r}'_{n+1},\ldots,\mathsf{r}'_{n+p}$ is of the form (3) and let $w'_i \stackrel{\mathsf{def}}{=} (\mathsf{r}'_1,i_1)\ldots(\mathsf{r}'_n,i_n)[(\mathsf{r}'_{n+1},i_{n+1})\ldots(\mathsf{r}'_{n+p},i_{n+p})]^i$, for all $i \geq 1$. By Lemma 8, there exist unfoldings $\mathsf{A}'(x_1,\ldots,x_{\#\mathsf{A}+1}) \overset{w'_i}{\Longrightarrow}_{\Delta'} \phi'_i$, stores ν_i and configurations γ_i, such that $\gamma_i \models^{\nu_i} \phi'_i$. We define:

$$\delta_i \stackrel{\mathsf{def}}{=} \|\{(c_1,p_1,\ldots,c_n,p_n) \in I_i \mid \nu(x_{\#\mathsf{A}+1}) = c_j, j \in [1,n]\}\|, \text{ for all } i \geq 1$$

where $\gamma_i \stackrel{\mathsf{def}}{=} (\mathcal{C}_i, I_i, \rho_i)$. Since $\gamma_i \models^{\nu_i} \phi'_i$ and one of the rules $\mathsf{r}'_{n+1},\ldots,\mathsf{r}'_{n+p}$ is of type (3), the sequence $\delta_1,\delta_2,\ldots$ is strictly increasing. Moreover, we have $\delta_i \leq \delta(\gamma_i)$, for all $i \geq 1$, hence there exists a sequence of integers $1 \leq i_1 < i_2 < \ldots$ such that $\delta(\gamma_{i_j}) < \delta(\gamma_{i_{j+1}})$, for all $j \geq 1$. □

The complexity result below uses a similar argument on the maximal size of (hence the number of) base tuples as in Theorem 1, leading to similar complexity gaps:

Theorem 3. $\mathsf{Bnd}^{(k,\infty)}[\Delta,\mathsf{A}]$ *is in* co-NP, $\mathsf{Bnd}^{(\infty,\ell)}[\Delta,\mathsf{A}]$ *is in* EXP *and* $\mathsf{Bnd}[\Delta,\mathsf{A}]$ *is in* 2EXP.

Proof. Lemma 11 shows the reduction of the complement of $\mathsf{Bnd}[\Delta,\mathsf{A}]$ to the existence of a reachable cycle in the graph $\mathcal{G}(\Delta')$, where Δ' is constructed from Δ in polynomial time. Moreover, we have $\text{arity}(\Delta') = \text{arity}(\Delta)+1$ and $\text{intersize}(\Delta') = \text{intersize}(\Delta)$. We distinguish the three cases below:

- $k < \infty$, $\ell = \infty$: in this case, we can define a non-deterministic algorithm as follows. We guess the solution $(\langle W_1,\ldots,W_K\rangle, W_{K+1}, \langle i_1,i_2,\ldots,i_n\rangle$, where:
  - $\langle W_1,\ldots,W_K\rangle$ defines an acyclic witness for a satisfiable least solution of A' in Δ' constructed as in the proof of Thm. 1;
  - $W_{K+1} = (T_{K+1}, r_{K+1}, t_{K+1,1}, ..., e_{K+1,h_{K+1}})$ is similar to a regular entry W_i, that is, contains a base tuple T_{K+1}, an index r_{K+1} of a rule of Δ' and indices $e_{K+1,1}$, $\ldots$, $e_{K+1,h_{K+1}} \in \{1,\ldots,K\}$ such that T_{K+1} is computed correctly by applying the rule r_{K+1} from base tuples $T_{e_{K+1,1}},\ldots,T_{e_{K+1,h_{K+1}}}$ as explained in the proof of Thm. 1;
  - $\langle i_1,i_2,\ldots,i_n\rangle$ defines an acyclic path starting at the initial node in the directed acyclic graph defined by W, that is, $1 = i_1 < i_2 < \ldots < i_n \leq K$ and moreover $i_{j+1} \in \{e_{i_j,1},\ldots,e_{i_j,h_{i_j}}\}$ for all $j \in \{1,2,\ldots,n-1\}$;
  - the path $\langle i_1,i_2,\ldots,i_n\rangle$ can be *closed* into a witness reachable cycle from i_1 by using W_{K+1} that is, whenever (i) rules r_{i_n} and r_K define the same predicate, and moreover $T_{i_n} = T_{K+1}$, (ii) the intersection $X = \{e_{K+1,1},\ldots,e_{K+1,h_{K+1}}\} \cap \{i_1,i_2,\ldots,i_n\} \neq \emptyset$, (iii) if $i_j = \min X$, that is, the cycle starts at i_j then at least one of the rules used along the cycle $r_{i_j}, r_{i_{j+1}},\ldots,r_{i_{n-1}}, r_{K+1}$ is of the form (3).

The solution is of linear size $O(\text{size}(\Delta'))$ by the same arguments as in the proof of Thm. 1. Therefore, it can be guessed in polynomial time, and moreover checked in polynomial time following the conditions above. This implies the membership of the complement problem in NP, henceforth $\mathsf{Bnd}^{(k,\infty)}[\Delta,\mathsf{A}]$ is in co-NP.

- $k=\infty$, $\ell<\infty$: in this case, using the algorithm from Fig. 3, the graph $\mathcal{G}(\Delta')$ is constructed in time $2^{poly(\text{size}(\Delta'))}$ as previously explained in the proof of Theorem 1. Finding a reachable cycle with the additional properties required by Lemma 11 can be done in two additional steps, respectively, first building the SCCs decomposition of $\mathcal{G}(\Delta')$ and then checking reachability of SCCs containing edges derived from rules of form (3) from SCCs containing vertices $(\mathsf{A}',\mathfrak{t})$. Both steps can be done in in linear time in the size of $\mathcal{G}(\Delta')$ i.e., using Tarjan algorithm for SCC decomposition and standard graph traversals. Therefore, the overall time complexity remains $2^{poly(\text{size}(\Delta'))}$, and as such $\mathsf{Bnd}^{(\infty,\ell)}[\Delta,\mathsf{A}]$ is in EXP.
- $k=\infty$, $\ell=\infty$: following the same argument as in the previous point and noticing that the graph $\mathcal{G}(\Delta')$ is constructed in time $2^{2^{poly(\text{size}(\Delta'))}}$ we conclude that $\mathsf{Bnd}[\Delta,\mathsf{A}]$ is in 2EXP. □

Moreover, the construction of $\mathcal{G}(\Delta')$ allows to prove the following cut-off result:

Proposition 1. *Let γ be a configuration and ν be a store, such that $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}})$. If $\mathsf{Bnd}^{(k,\ell)}[\Delta,\mathsf{A}]$ then (1) $\delta(\gamma)=poly(\text{size}(\Delta))$ if $k<\infty$, $\ell=\infty$, (2) $\delta(\gamma)=2^{poly(\text{size}(\Delta))}$ if $k=\infty$, $\ell<\infty$ and (3) $\delta(\gamma)=2^{2^{poly(\text{size}(\Delta))}}$ if $k=\infty$, $\ell=\infty$.*

Proof. First, we show that in all cases, the degree is bounded by $2^{B^*}\cdot L\cdot I$ where B^* is the maximal length of a satisfiable base tuple in Δ', L is the number of predicates in Δ' and I is the maximal number of interactions defined in a rule in Δ'. The maximal length B^* of a satisfiable base tuples has been considered in the proof of Thm. 1 to derive an upper bound on the the number of distinct satisfiable base tuples for a SID. Then, $2^{B^*}\cdot L$ represents a bound on the number of nodes in the graph $\mathcal{G}(\Delta')$ as for every predicate there will be at most 2^{B^*} satisfiable base tuples associated to it. Meantime, this value also represents a bound on the longest acyclic path in $\mathcal{G}(\Delta')$. We are interested on acyclic paths because cycles in $\mathcal{G}(\Delta')$ are guaranteed to never connect (use in interactions) the extra variable introduced in Δ' (otherwise the system would not be of bounded degree). But then, along the acyclic paths, at most I interactions are defined at each step, henceforth, the bound of $2^{B^*}\cdot L\cdot I$ on the number on total interactions that could involve the extra variable.

Second, let observe that both L and I are the same in Δ' and in Δ and equal to $O(\text{size}(\Delta))$. Moreover, it was shown in the proof of Thm. 1 that $B^*=2\alpha+2\alpha^2+p^{\min(\alpha,\beta)}\alpha^{\min(\alpha,\beta)}$ for $\alpha=\text{arity}(\Delta')=\text{arity}(\Delta)+1$ and $\beta=\text{intersize}(\Delta')=\text{intersize}(\Delta)$, $p=\|\mathcal{P}\|$ the number of ports. Henceforth, we distinguished the three cases, respectively (i) $B^*=O(1)$ if $k<\infty$, $\ell=\infty$, (ii) $B^*=poly(\text{size}(\Delta))$ if $k=\infty$, $\ell<\infty$ and (iii) $B^*=2^{poly(\text{size}(\Delta))}$ if $k=\infty$, $\ell=\infty$. By using the above in the expression $2^{B^*}\cdot L\cdot I$ we obtain the values of the bound as stated in the Proposition. □

6 Entailment

This section is concerned with the entailment problem $\mathsf{Entl}[\Delta, \mathsf{A}, \mathsf{B}]$, that asks whether $\gamma \models^{\nu}_{\Delta} \exists x_{\#\mathsf{A}+1} \ldots \exists x_{\#\mathsf{B}} \,.\, \mathsf{B}(x_1, \ldots, x_{\#\mathsf{B}})$, for every configuration γ and store ν, such that $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$. For instance, the proof from Fig. 1 (c) relies on the following entailments, that occur as the side conditions of the Hoare logic rule of consequence:

$$\mathsf{ring}_{h,t}(y) \models_{\Delta} \exists x \exists z.[y]@\mathsf{H} * \langle y.out, z.in \rangle * \mathsf{chain}_{h-1,t}(z,x) * \langle x.out, y.in \rangle$$
$$[z]@\mathsf{H} * \langle z.out, x.in \rangle * \mathsf{chain}_{h-1,t}(x,y) * \langle y.out, z.in \rangle \models_{\Delta} \mathsf{ring}_{h,t}(z)$$

By introducing two fresh predicates A_1 and A_2, defined by the rules:

$$\mathsf{A}_1(x_1) \leftarrow \exists y \exists z.[x_1]@\mathsf{H} * \langle x_1.out, z.in \rangle * \mathsf{chain}_{h-1,t}(z,y) * \langle y.out, x_1.in \rangle \quad (4)$$
$$\mathsf{A}_2(x_1,x_2) \leftarrow \exists z.[x_1]@\mathsf{H} * \langle x_1.out, z.in \rangle * \mathsf{chain}_{h-1,t}(z,x_2) * \langle x_2.out, x_1.in \rangle \quad (5)$$

the above entailments are equivalent to $\mathsf{Entl}[\Delta, \mathsf{ring}_{h,t}, \mathsf{A}_1]$ and $\mathsf{Entl}[\Delta, \mathsf{A}_2, \mathsf{ring}_{h,t}]$, respectively, where Δ consists of the rules (4) and (5), together with the rules that define the $\mathsf{ring}_{h,t}$ and $\mathsf{chain}_{h,t}$ predicates (§1.1).

We show that the entailment problem is undecidable, in general (Thm. 4), and recover a decidable fragment, by means of three syntactic conditions, typically met in our examples. These conditions use the following notion of *profile*:

Definition 11. *The* profile *of a SID Δ is the pointwise greatest function $\lambda_{\Delta} : \mathbb{A} \to \mathrm{pow}(\mathbb{N})$, mapping each predicate A into a subset of $[1, \#\mathsf{A}]$, such that, for each rule $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \phi$ from Δ, each atom $\mathsf{B}(y_1, \ldots, y_{\#\mathsf{B}})$ from ϕ and each $i \in \lambda_{\Delta}(\mathsf{B})$, there exists $j \in \lambda_{\Delta}(\mathsf{A})$, such that x_j and y_i are the same variable.*

The profile identifies the parameters of a predicate that are always replaced by a variable $x_1, \ldots, x_{\#\mathsf{A}}$ in each unfolding of $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$, according to the rules in Δ; it is computed by a greatest fixpoint iteration, in time $poly(\mathrm{size}(\Delta))$.

Definition 12. *A rule $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \phi * \mathbin{\text{⫽}}_{\ell=1}^{h} \mathsf{B}_{\ell}(z_1^{\ell}, \ldots, z_{\#\mathsf{B}_{\ell}}^{\ell})$, where ϕ is a quantifier- and predicate-free formula, is said to be:*

1. progressing *if and only if $\phi = [x_1] * \psi$, where ψ consists of interaction atoms involving x_1 and (dis-)equalities, such that $\bigcup_{\ell=1}^{h} \{z_1^{\ell}, \ldots, z_{\#\mathsf{B}_{\ell}}^{\ell}\} = \{x_2, \ldots, x_{\#\mathsf{A}}\} \cup \{y_1, \ldots, y_m\}$,*
2. connected *if and only if, for each $\ell \in [1,h]$ there exists an interaction atom in ψ that contains both z_1^{ℓ} and a variable from $\{x_1\} \cup \{x_i \mid i \in \lambda_{\Delta}(\mathsf{A})\}$,*
3. equationally-restricted (e-restricted) *if and only if, for every disequation $x \neq y$ from ϕ, we have $\{x,y\} \cap \{x_i \mid i \in \lambda_{\Delta}(\mathsf{A})\} \neq \emptyset$.*

A SID Δ is progressing, connected *and* e-restricted *if and only if each rule in Δ is* progressing, connected *and* e-restricted, *respectively.*

For example, the SID consisting of the rules from §1.1, together with rules (4) and (5) is progressing, connected and e-restricted. For a configuration $\gamma = (\mathcal{C}, I, \rho)$, let:

$$\mathrm{nodes}(\gamma) \stackrel{\mathsf{def}}{=} \mathcal{C} \cup \{c_i \mid (c_1, p_1, \ldots, c_n, p_n) \in I, i \in [1,n]\}$$

be the set of (possibly absent) components that occur in γ.

Lemma 12. *Given a progressing SID* Δ *and a predicate* $\mathsf{A} \in \mathrm{def}(\Delta)$*, for any configuration* $\gamma = (C, I, \rho)$ *and store* ν*, such that* $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$*, we have* $\{\nu(x_1), \ldots, \nu(x_{\#\mathsf{A}})\} \subseteq \mathrm{nodes}(\gamma) = C$.

Proof. We proceed by fixpoint induction on the definition of $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$. By definition, there exists a progressing rule

$$\mathsf{r} \;:\; \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, [x_1] * \psi * \mathop{\raisebox{-.5ex}{\Large$\ast$}}_{\ell=1}^{h} \mathsf{B}_\ell(z^\ell_1, \ldots, z^\ell_{\#\mathsf{B}_\ell})$$

a store ν' and configurations $\gamma_0, \ldots, \gamma_h$ such that:

- $\gamma = \gamma_0 \bullet \ldots \bullet \gamma_h$,
- $\nu(x_i) = \nu'(x_i)$ for all $i \in [1, \#\mathsf{A}]$,
- $\gamma_0 \models^{\nu'}_{\Delta} [x_1] * \psi$, and
- $\gamma_\ell \models^{\nu'}_{\Delta} \mathsf{B}_\ell(z^\ell_1, \ldots, z^\ell_{\#\mathsf{B}_\ell})$ for all $\ell \in [1, h]$.

For $1 \leq \ell \leq h$, let $\nu_\ell(x_i) = \nu'(z^\ell_i)$ for $1 \leq i \leq \#\mathsf{B}_\ell$. Now apply the induction hypothesis on the derivation of $\gamma_\ell \models^{\nu_\ell}_{\Delta} \mathsf{B}_\ell(x_1, \ldots, x_{\#\mathsf{B}_\ell})$ to obtain that $\{\nu_\ell(x_1), \ldots, \nu_\ell(x_{\#\mathsf{B}_\ell})\} \subseteq \mathrm{nodes}(\gamma_\ell)$. Since r is progressing, we have:

$$\begin{aligned}
&\{\nu(x_1), \ldots, \nu(x_{\#\mathsf{A}})\} \subseteq \{\nu'(x_1), \ldots, \nu'(x_{\#\mathsf{A}})\} \cup \{\nu'(y_1), \ldots, \nu'(y_m)\} \\
&= \{\nu'(x_1)\} \cup \bigcup_{\ell=1}^{h} \{\nu'(z^\ell_1), \ldots, \nu'(z^\ell_{\#\mathsf{B}_\ell})\} = \{\nu'(x_1)\} \cup \bigcup_{\ell=1}^{h} \{\nu_\ell(x_1), \ldots, \nu_\ell(x_{\#\mathsf{B}_\ell})\} \\
&\subseteq \mathrm{nodes}(\gamma_0) \cup \bigcup_{\ell=1}^{h} \mathrm{nodes}(\gamma_\ell) = \mathrm{nodes}(\gamma) \quad \square
\end{aligned}$$

We recall that $\mathrm{def}_\Delta(\mathsf{A})$ is the set of rules from Δ that define A and denote by $\mathrm{def}^*_\Delta(\mathsf{A})$ the least superset of $\mathrm{def}_\Delta(\mathsf{A})$ containing the rules that define a predicate from a rule in $\mathrm{def}^*_\Delta(\mathsf{A})$. The following result shows that the entailment problem becomes undecidable as soon as the connectivity condition is even slightly lifted:

Theorem 4. $\mathsf{Entl}[\Delta, \mathsf{A}, \mathsf{B}]$ *is undecidable, even when* Δ *is progressing and e-restricted, and only the rules in* $\mathrm{def}^*_\Delta(\mathsf{A})$ *are connected (the rules in* $\mathrm{def}^*_\Delta(\mathsf{B})$ *may be disconnected).*

Proof. By a reduction from the known undecidable problem of universality of context-free languages [3]. A context-free grammar $G = \langle N, T, S, \Delta \rangle$ consists of a finite set N of nonterminals, a finite set T of terminals, a start symbol $S \in N$ and a finite set Δ of productions of the form $A \to w$, where $A \in N$ and $w \in (N \cup T)^*$. Given finite strings $u, v \in (N \cup T)^*$, the step relation $u \Rightarrow v$ replaces a nonterminal A of u by the right-hand side w of a production $A \to w$ and $\Rightarrow^*$ denotes the reflexive and transitive closure of $\Rightarrow$. The language of G is the set $\mathcal{L}(G)$ of finite strings $w \in T^*$, such that $s \Rightarrow^* w$. The problem $T^* \subseteq \mathcal{L}(G)$ is known as the universality problem, known to be undecidable. Moreover, we assume w.l.o.g. that:

- $T = \{0, 1\}$, because every terminal can be encoded as a binary string,
- $\mathcal{L}(G)$ does not contain the empty string ε, because computing a grammar G' such that $\mathcal{L}(G') = \mathcal{L}(G) \cap T^+$ is possible and, moreover, we can reduce from the modified universality problem problem $T^+ \subseteq \mathcal{L}(G')$ instead of the original $T^* \subseteq \mathcal{L}(G)$,

– G is in Greibach normal form, i.e. it contains only production rules of the form $\mathsf{B}_0 \to b\mathsf{B}_1 \ldots \mathsf{B}_n$, where $\mathsf{B}_0, \ldots \mathsf{B}_n \in N$, for some $n \geq 0$ and $b \in T$.

Let $\mathcal{P} = \{p_0, p_1\}$ be a set of ports. For each nonterminal $\mathsf{B}_0 \in N$, we have a predicate B_0 or arity two and a rule $\mathsf{B}_0(x_1, x_2) \leftarrow \exists y_1 \ldots \exists y_n \ . \ [x_1] * \langle x_1.p_a, y_1.p_a \rangle * \mathsf{B}_1(y_1, y_2) * \ldots * \mathsf{B}_n(y_n, x_2)$, for each rule $\mathsf{B}_0 \to b\mathsf{B}_1 \ldots \mathsf{B}_n$ of G. Moreover, we consider the rules $\mathsf{A}(x_1, x_2) \leftarrow \exists z \ . \ \langle x_1.p_a, z.p_a \rangle * \mathsf{A}(z, x_2)$ and $\mathsf{A}(x_1, x_2) \leftarrow \langle x_1.p_a, x_2.p_a \rangle$, for all $a \in \{0, 1\}$. Let Δ be the SID containing the above rules. It is easy to check that the SID is progressing and e-restricted and that, moreover, the rules from $\mathsf{def}^*_\Delta(\mathsf{A})$ are connected. Finally, $\mathsf{A}(x_1, x_2) \models_\Delta \mathsf{B}(x_1, x_2)$ if and only if $T^+ \subseteq \mathcal{L}(G)$. □

On the positive side, we prove that $\mathsf{Entl}[\Delta, \mathsf{A}, \mathsf{B}]$ is decidable, if Δ is progressing, connected and e-restricted, assuming further that $\mathsf{Bnd}[\Delta, \mathsf{A}]$ has a positive answer. In this case, the bound on the degree of the models of $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$ is effectively computable, using the algorithm from Fig. 3 (see Prop. 1 for a cut-off result) and denote by $\mathfrak{B}$ this bound, throughout this section.

The proof uses a reduction of $\mathsf{Entl}[\Delta, \mathsf{A}, \mathsf{B}]$ to a similar problem for SL, showed to be decidable [20]. We recall the definition of SL, interpreted over heaps $\mathsf{h} : \mathbb{C} \rightharpoonup_{fin} \mathbb{C}^{\mathfrak{K}}$, introduced in §2.3. SL rules are denoted as $\overline{\mathsf{A}}(x_1, \ldots, x_{\#(\overline{\mathsf{A}})}) \leftarrow \phi$, where ϕ is a SL formula, such that $\mathsf{fv}(\phi) \subseteq \{x_1, \ldots, x_{\#(\overline{\mathsf{A}})}\}$ and SL SIDs are denoted as $\overline{\Delta}$. The profile $\lambda_{\overline{\Delta}}$ is defined for SL same as for CL (Def. 11).

Definition 13. *A SL rule $\overline{\mathsf{A}}(x_1, \ldots, x_{\#(\overline{\mathsf{A}})}) \leftarrow \phi$ from a SID $\overline{\Delta}$ is said to be:*

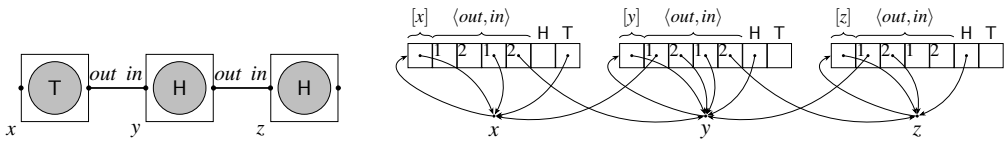
1. progressing *if and only if $\phi = \exists t_1 \ldots \exists t_m \ . \ x_1 \mapsto (y_1, \ldots, y_{\mathfrak{K}}) * \psi$, where ψ contains only predicate and equality atoms,*
2. connected *if and only if $z_1 \in \{x_i \mid i \in \lambda_{\overline{\Delta}}(\overline{\mathsf{A}})\} \cup \{y_1, \ldots, y_{\mathfrak{K}}\}$, for every predicate atom $\overline{\mathsf{B}}(z_1, \ldots, z_{\#(\overline{\mathsf{B}})})$ from ϕ.*

Note that the definitions of progressing and connected rules are different for SL, compared to CL (Def. 12); in the rest of this section, we rely on the context to distinguish progressing (connected) SL rules from progressing (connected) CL rules. Moreover, e-restricted rules are defined in the same way for CL and SL (point 3 of Def. 12). A tight upper bound on the complexity of the entailment problem between SL formulæ, interpreted by progressing, connected and e-restricted SIDs, is given below:

Theorem 5 ([20]). *The SL entailment problem is in $2^{2^{poly(\mathsf{width}(\overline{\Delta}) \cdot \log \mathsf{size}(\overline{\Delta}))}}$, for progressing, connected and e-restricted SIDs.*

The reduction of $\mathsf{Entl}[\Delta, \mathsf{A}, \mathsf{B}]$ to SL entailments is based on the idea of viewing a configuration as a logical structure (hypergraph), represented by an indirected *Gaifman graph*, in which every tuple from a relation (hyperedge) becomes a clique [26]. In a similar vein, we encode a configuration, of degree at most $\mathfrak{B}$, by a heap of degree $\mathfrak{K}$ (Def. 14), such that $\mathfrak{K}$ is defined using the following integer function:

$$\mathrm{pos}(i, j, k) \stackrel{\mathsf{def}}{=} 1 + \mathfrak{B} \cdot \sum_{\ell=1}^{j-1} |\tau_\ell| + i \cdot |\tau_j| + k$$



(a) (b)

Fig. 4: Gaifman Heap for a Chain Configuration

where $\mathsf{Inter} \stackrel{\text{def}}{=} \{\tau_1, \ldots, \tau_M\}$ is the set of interaction types and $Q \stackrel{\text{def}}{=} \{q_1, \ldots, q_N\}$ is the set of states of the behavior $\mathbb{B} = (\mathcal{P}, Q, \rightarrow)$ (§2). Here $i \in [0, \mathfrak{B}-1]$ denotes an interaction of type $j \in [1, M]$ and $k \in [0, N-1]$ denotes a state. We use M and N throughout the rest of this section, to denote the number of interaction types and states, respectively.

For a set I of interactions, let $\mathsf{Tuples}_I^j(c) \stackrel{\text{def}}{=} \{\langle c_1, \ldots, c_n\rangle \mid (c_1, p_1, \ldots, c_n, p_n) \in I,\ \tau_j = \langle p_1, \ldots, p_n\rangle,\ c \in \{c_1, \ldots, c_n\}\}$ be the tuples of components from an interaction of type τ_j from I, that contain a given component c.

Definition 14. *Given a configuration $\gamma = (\mathcal{C}, I, \rho)$, such that $\delta(\gamma) \leq \mathfrak{B}$, a* Gaifman heap *for γ is a heap $\mathsf{h} : \mathbb{C} \rightharpoonup_{fin} \mathbb{C}^{\mathfrak{K}}$, where $\mathfrak{K} \stackrel{\text{def}}{=} \text{pos}(0, M+1, N)$, $\text{dom}(\mathsf{h}) = \text{nodes}(\gamma)$ and, for all $c_0 \in \text{dom}(\mathsf{h})$, such that $\mathsf{h}(c_0) = \langle c_1, \ldots, c_{\mathfrak{K}}\rangle$, the following hold:*

1. *$c_1 = c_0$ if and only if $c_0 \in \mathcal{C}$,*
2. *for all $j \in [1, M]$, $\mathsf{Tuples}_I^j(c) = \{\mathbf{c}_1, \ldots, \mathbf{c}_s\}$ if and only if there exist integers $0 \leq k_1 < \ldots < k_s < \mathfrak{B}$, such that $\langle \mathsf{h}(c_0)\rangle_{\text{inter}(k_i, j)} = \mathbf{c}_i$, for all $i \in [1, s]$, where $\text{inter}(i, j) \stackrel{\text{def}}{=} [\text{pos}(i-1, j, 0), \text{pos}(i, j, 0)]$ are the entries of the i-th interaction of type τ_j in $\mathsf{h}(c_0)$,*
3. *for all $k \in [1, N]$, we have $\langle \mathsf{h}(c_0)\rangle_{\text{state}(k)} = c_0$ if and only if $\rho(c_0) = q_k$, where the entry $\text{state}(k) \stackrel{\text{def}}{=} \text{pos}(0, M+1, k-1)$ in $\mathsf{h}(c_0)$ corresponds to the state $q_k \in Q$,*

We denote by $\mathbb{G}(\gamma)$ the set of Gaifman heaps for γ.

Intuitively, if h is a Gaifman heap for γ and $c_0 \in \text{dom}(\mathsf{h})$, then the first entry of $\mathsf{h}(c_0)$ indicates whether c_0 is present (condition 1 of Def. 14), the next $\mathfrak{B} \cdot \sum_{j=1}^{M} |\tau_j|$ entries are used to encode the interactions of each type τ_j (condition 2 of Def. 14), whereas the last N entries are used to represent the state of the component (condition 3 of Definition 14). Note that the encoding of configurations by Gaifman heaps is not unique: two Gaifman heaps for the same configuration may differ in the order of the tuples from the encoding of an interaction type and the choice of the unconstrained entries from $\mathsf{h}(c_0)$, for each $c_0 \in \text{dom}(\mathsf{h})$. On the other hand, if two configurations have the same Gaifman heap encoding, they must be the same configuration.

Example 4. Fig. 4 (b) shows a Gaifman heap for the configuration in Fig. 4 (a), where each component belongs to at most 2 interactions of type $\langle out, in\rangle$. ■

We say that a configuration γ' is a *subconfiguration* of γ, denoted $\gamma' \sqsubseteq \gamma$ if and only if $\gamma = \gamma' \bullet \gamma''$, for some configuration γ''. The following lemma builds Gaifman heaps for subconfigurations:

Lemma 13. *Given configurations γ and γ', such that $\gamma' \sqsubseteq \gamma$, if $\mathsf{h} \in \mathbb{G}(\gamma)$, then $\mathsf{h}' \in \mathbb{G}(\gamma')$, where $\text{dom}(\mathsf{h}') = \text{dom}(\mathsf{h}) \cap \text{nodes}(\gamma')$ and $\mathsf{h}'(c) = \mathsf{h}(c)$, for all $c \in \text{dom}(\mathsf{h}')$.*

Proof. We have $\mathrm{dom}(\mathsf{h}') = \mathrm{dom}(\mathsf{h}) \cap \mathrm{nodes}(\gamma') = \mathrm{nodes}(\gamma) \cap \mathrm{nodes}(\gamma') = \mathrm{nodes}(\gamma')$, because $\mathrm{dom}(\mathsf{h}) = \mathrm{nodes}(\gamma) \supseteq \mathrm{nodes}(\gamma')$. The points (1-3) of Def. 14 are by easy inspection. $\square$

We build a SL SID $\overline{\Delta}$ that generates the Gaifman heaps of the models of the predicate atoms occurring in a progressing CL SID Δ. The construction associates to each variable x, that occurs free or bound in a rule from Δ, a unique $\mathfrak{K}$-tuple of variables $\eta(x) \in \mathbb{V}^{\mathfrak{K}}$, that represents the image of the store value $\nu(x)$ in a Gaifman heap h i.e., $\mathsf{h}(\nu(x)) = \nu(\eta(x))$. Moreover, we consider, for each predicate symbol $\mathsf{A} \in \mathrm{def}(\Delta)$, an annotated predicate symbol $\overline{\mathsf{A}}_\iota$ of arity $\#\overline{\mathsf{A}}_\iota = (\mathfrak{K}+1) \cdot \#\mathsf{A}$, where $\iota : [1, \#\mathsf{A}] \times [1, M] \to 2^{[0, \mathfrak{B}-1]}$ is a map associating each parameter $i \in [1, \#\mathsf{A}]$ and each interaction type τ_j, for $j \in [1, M]$, a set of integers $\iota(i, j)$ denoting the positions of the encodings of the interactions of type τ_j, involving the value of x_i, in the models of $\overline{\mathsf{A}}_\iota(x_1, \ldots, x_{\#\mathsf{A}}, \eta(x_1), \ldots, \eta(x_{\#\mathsf{A}}))$ (point 2 of Def. 14). Then $\overline{\Delta}$ contains rules of the form:

$$\overline{\mathsf{A}}_\iota(x_1, \ldots, x_{\#(\mathsf{A})}, \eta(x_1), \ldots, \eta(x_{\#(\mathsf{A})})) \leftarrow \tag{6}$$
$$\exists y_1 \ldots \exists y_m \exists \eta(y_1) \ldots \exists \eta(y_m) \;.\; \overline{\psi} * \pi * \circledast_{\ell=1}^{h} \overline{\mathsf{B}}^{\ell}_{\iota^\ell}(z^\ell_1, \ldots, z^\ell_{\#(\mathsf{B}^\ell)}, \eta(z^\ell_1), \ldots, \eta(z^\ell_{\#(\mathsf{B}^\ell)}))$$

for which Δ has a *stem rule* $\mathsf{A}(x_1, \ldots, x_{\#(\mathsf{A})}) \leftarrow \exists y_1 \ldots \exists y_m \;.\; \psi * \pi * \circledast_{\ell=1}^{h} \mathsf{B}^\ell(z^\ell_1, \ldots, z^\ell_{\#\mathsf{B}^\ell})$, where $\psi * \pi$ is a quantifier- and predicate-free formula and π is the conjunction of equalities and disequalities from $\psi * \pi$. However, not all rules (6) are considered in $\overline{\Delta}$, but only the ones meeting the following condition:

Definition 15. *A rule of the form (6) is* well-formed *if and only if, for each $i \in [1, \#\mathsf{A}]$ and each $j \in [1, M]$, there exists a set of integers $Y_{i,j} \subseteq [0, \mathfrak{B}-1]$, such that:*

- $\|Y_{i,j}\| = \|I^j_{\psi,\pi}(x_i)\|$, *where $I^j_{\psi,\pi}(x)$ is the set of interaction atoms $\langle z_1.p_1, \ldots, z_n.p_n \rangle$ from ψ of type $\tau_j = \langle p_1, \ldots, p_n \rangle$, such that $z_s \approx_\pi x$, for some $s \in [1, n]$,*
- $Y_{i,j} \subseteq \iota(i, j)$ *and* $\iota(i, j) \setminus Y_{i,j} = \mathcal{Z}_j(x_i)$, *where* $\mathcal{Z}_j(x) \stackrel{\mathsf{def}}{=} \bigcup_{\ell=1}^{h} \bigcup_{k=1}^{\#\mathsf{B}^\ell} \{\iota^\ell(k, j) \mid x \approx_\pi z^\ell_k\}$ *is the set of positions used to encode the interactions of type τ_j involving the store value of the parameter x, in the sub-configuration corresponding to an atom $\mathsf{B}_\ell(z^\ell_1, \ldots, z^\ell_{\#(\mathsf{B}^\ell)})$, for some $\ell \in [1, h]$.*

We denote by $\overline{\Delta}$ the set of well-formed rules (6), such that, moreover:

$$\overline{\psi} \stackrel{\mathsf{def}}{=} x_1 \mapsto \eta(x_1) \;*\; \circledast_{x \in \mathrm{fv}(\psi)} \mathsf{CompStates}_\psi(x) \;*\; \circledast_{i=1}^{\#\mathsf{A}} \mathsf{InterAtoms}_\psi(x_i), \text{ where:}$$
$$\mathsf{CompStates}_\psi(x) \stackrel{\mathsf{def}}{=} \circledast_{[x] \text{ occurs in } \psi} \langle \eta(x) \rangle_1 = x \;*\; \circledast_{x@q_k \text{ occurs in } \psi} \langle \eta(x) \rangle_{\mathrm{state}(k)} = x$$
$$\mathsf{InterAtoms}_\psi(x_i) \stackrel{\mathsf{def}}{=} \circledast_{j=1}^{M} \circledast_{p=1}^{r_j} \langle \eta(x_i) \rangle_{\mathrm{inter}(j, k^j_p)} = \mathbf{x}^j_p \text{ and } \{k^j_1, \ldots, k^j_{r_j}\} \stackrel{\mathsf{def}}{=} \iota(i, j) \setminus \mathcal{Z}_j(x_i)$$

Here for two tuples of variables $\mathbf{x} = \langle x_1, \ldots, x_k \rangle$ and $\mathbf{y} = \langle y_1, \ldots, y_k \rangle$, we denote by $\mathbf{x} = \mathbf{y}$ the formula $\circledast_{i=1}^{k} x_i = y_i$. Intuitively, the SL formula $\mathsf{CompStates}_\psi(x)$ realizes the encoding of the component and state atoms from ψ, in the sense of points (1) and (3) from Def. 14, whereas the formula $\mathsf{InterAtoms}_\psi(x_i)$ realizes the encodings of the interactions involving a parameter x_i in the stem rule (point 2 of Def. 14). In particular, the definition of $\mathsf{InterAtoms}_\psi(x_i)$ uses the fact that the rule is well-formed.

Lemma 14. *Let Δ be a progressing SID and $\mathsf{A} \in \mathrm{def}(\Delta)$ be a predicate, such that $\gamma \models^{\nu} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$, for some configuration $\gamma = (\mathcal{C}, I, \rho)$ and store ν. Then, for each heap $\mathsf{h} \in \mathbb{G}(\gamma)$, there exists a map $\iota : [1, \#\mathsf{A}] \times [1, M] \to 2^{[0, \mathfrak{B}-1]}$ and a store $\overline{\nu}$, such that the following hold:*

1. $\overline{\nu}(x_i) = \nu(x_i) \in \mathrm{dom}(\mathsf{h})$ *and* $\overline{\nu}(\eta(x_i)) = \mathsf{h}(\nu(x_i))$, $\forall i \in [1, \#\mathsf{A}]$,
2. $\mathsf{Tuples}^{j}_{I}(\overline{\nu}(x_i)) = \left\{ \langle \overline{\nu}(\eta(x_i)) \rangle_{\mathrm{inter}(j,k)} \mid k \in \iota(i,j) \right\}$, $\forall i \in [1, \#\mathsf{A}]\ \forall j \in [1, M]$,
3. $\mathsf{h} \Vdash^{\overline{\nu}}_{\Delta} \overline{\mathsf{A}}_{\iota}(x_1, \ldots, x_{\#\mathsf{A}}, \eta(x_1), \ldots, \eta(x_{\#(\mathsf{A})}))$.

Proof. By induction on the definition of $\gamma \models^{\nu} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$, assume that $\gamma \models^{\nu} \exists y_1 \ldots \exists y_m \,.\, \psi * \pi * \circledast_{\ell=1}^{h} \mathsf{B}^{\ell}(z^{\ell}_1, \ldots, z^{\ell}_{\#(\mathsf{B}^{\ell})})$, where $\mathsf{A}(x_1, \ldots, x_{\#(\mathsf{A})}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \psi * \pi * \circledast_{\ell=1}^{h} \mathsf{B}^{\ell}(z^{\ell}_1, \ldots, z^{\ell}_{\#(\mathsf{B}^{\ell})})$ is a rule from Δ, such that $\psi * \pi$ is quantifier- and predicate-free and π is the conjunction of equalities and disequalities from $\psi * \pi$. Then there exists a store ν', that agrees with ν over $x_1, \ldots, x_{\#\mathsf{A}}$, and configurations $\gamma_0 = (\mathcal{C}_0, I_0, \rho), \ldots, \gamma_h = (\mathcal{C}_h, I_h, \rho)$, such that:

- $\gamma_0 \models^{\nu'}_{\Delta} \psi * \pi$,
- $\gamma_{\ell} \models^{\nu'}_{\Delta} \mathsf{B}_{\ell}(z^{\ell}_1, \ldots, z^{\ell}_{\#\mathsf{B}_{\ell}})$, for all $\ell \in [1, h]$, and
- $\gamma = \gamma_0 \bullet \ldots \bullet \gamma_h$.

We define the heaps $\mathsf{h}_0, \ldots, \mathsf{h}_h$, as follows:

- for each $\ell \in [1, h]$, let $\mathrm{dom}(\mathsf{h}_{\ell}) = \mathrm{nodes}(\gamma_{\ell})$, $\mathsf{h}_{\ell}(c) = \mathsf{h}(c)$, for all $c \in \mathrm{dom}(\mathsf{h}_{\ell})$,
- $\mathsf{h}_0 \stackrel{\mathsf{def}}{=} \mathsf{h} \setminus (\bigcup_{\ell=1}^{h} \mathsf{h}_{\ell})$.

By Lemma 13, we obtain that $\mathsf{h}_{\ell} \in \mathbb{G}(\gamma_{\ell})$, for all $\ell \in [1, h]$. We define $\mathsf{h} \stackrel{\mathsf{def}}{=} \mathsf{h}_0 \cup \ldots \cup \mathsf{h}_h$ and prove that this is indeed a heap, by showing $\mathrm{dom}(\mathsf{h}_i) \cap \mathrm{dom}(\mathsf{h}_j) = \emptyset$, for all $0 \leq i < j \leq h$. If $i = 0$, we have $\mathrm{dom}(\mathsf{h}_i) \cap \mathrm{dom}(\mathsf{h}_j) = \emptyset$, by the definition of h_i. Else, suppose, for a contradiction, that $c \in \mathrm{dom}(\mathsf{h}_i) \cap \mathrm{dom}(\mathsf{h}_j)$, for some $1 \leq i < j \leq h$. Then $c \in \mathrm{nodes}(\gamma_i) \cap \mathrm{nodes}(\gamma_j)$. Since $\gamma_{\ell} \models \exists x_1 \ldots \exists x_{\#\mathsf{B}_{\ell}} \,.\, \mathsf{B}(x_1, \ldots, x_{\#\mathsf{B}_{\ell}})$, by Lemma 12, we obtain $c \in \mathcal{C}_i \cap \mathcal{C}_j$, which contradicts the fact that $\gamma_i \bullet \gamma_j$ is defined. Next, we apply the inductive hypothesis to find stores $\overline{\nu}_{\ell}$ and maps ι^{ℓ} such that, for $\ell \in [1, h]$, we have:

- $\overline{\nu}(z^{\ell}_i) = \nu'(z^{\ell}_i) \in \mathrm{dom}(\mathsf{h}_{\ell})$ and $\mathsf{h}_{\ell}(\nu'(z^{\ell}_i)) = \overline{\nu}(\eta(z^{\ell}_i))$, $\forall i \in [1, \#\mathsf{B}_{\ell}]$,
- $\mathsf{Tuples}^{j}_{I_{\ell}}(\overline{\nu}(z^{\ell}_i)) = \left\{ \langle \overline{\nu}_{\ell}(\eta(z^{\ell}_i)) \rangle_{\mathrm{inter}(j,k)} \mid k \in \iota^{\ell}(i,j) \right\}$, $\forall i \in [1, \#(\mathsf{B}_{\ell})]\ \forall j \in [1, M]$, and
- $\mathsf{h}_{\ell} \Vdash^{\overline{\nu}_{\ell}}_{\Delta} \overline{\mathsf{B}_{\ell}}_{\iota}(z^{\ell}_1, \ldots, z^{\ell}_{\#\mathsf{B}_{\ell}}, \eta(z^{\ell}_1), \ldots, \eta(z^{\ell}_{\#(\mathsf{B}_{\ell})}))$.

First, for each $i \in [1, \#\mathsf{A}]$ and each $j \in [1, M]$, we define $\iota(i,j) \stackrel{\mathsf{def}}{=} \{k_1, \ldots, k_{s_i}\} \cup \bigcup_{\ell=1}^{h} \bigcup_{k=1}^{\#\mathsf{B}_{\ell}} \left\{ \iota^{\ell}(k,j) \mid x_i \approx_{\pi} z^{\ell}_k \right\}$, where:

- $\mathsf{Tuples}^{j}_{I_0}(\nu(x_i)) = \{\mathbf{c}_1, \ldots, \mathbf{c}_{s_{i,j}}\}$, and
- $0 \leq k^{i,j}_1 < \ldots < k^{i,j}_{s_{i,j}} < \mathfrak{B}$ are integers, such that $\langle \mathsf{h}(\nu(x_i)) \rangle_{\mathrm{inter}(j, k^{i,j}_{\ell})} = \mathbf{c}_{\ell}$, $\forall \ell \in [1, s_{i,j}]$; the existence of these integers is stated by point (2) of Def. 14, relative to $\nu(x_i)$.

Second, we define the store $\overline{\nu}$ as follows:

- $\overline{\nu}(x_1) = \nu(x_1)$ and $\overline{\nu}(\eta(x_1)) \stackrel{\mathsf{def}}{=} \mathsf{h}(\nu(x_1))$,
- $\overline{\nu}(z^{\ell}_i) \stackrel{\mathsf{def}}{=} \nu'(z^{\ell}_i)$, $\forall \ell \in [1, h]\ \forall i \in [1, \#\mathsf{B}_{\ell}]$,
- $\overline{\nu}(\eta(z^{\ell}_i)) \stackrel{\mathsf{def}}{=} \mathsf{h}(\nu'(z^{\ell}_i))$, $\forall \ell \in [1, h]\ \forall i \in [1, \#\mathsf{B}_{\ell}]$,
- $\overline{\nu}$ is arbitrary everywhere else.

The points (1) and (2) of the statement follow from the definitions of $\overline{\nu}$ and ι, respectively. To prove point (3), suppose, for a contradiction, that $k \in \{k^{i,j}_1, \ldots, k^{i,j}_{s_{i,j}}\} \cap \iota^{\ell}(t,j) \neq$

$\emptyset$, for some $i \in [1,\#\mathsf{A}]$, $j \in [1,M]$, $t \in [1,\#\mathsf{B}_\ell]$ and $\ell \in [1,h]$, such that $x_i \approx_\pi z_t^\ell$. Then there exists a tuple of components $\mathbf{c} \in \mathsf{Tuples}^j_{I_0}(\nu(x_i))$, such that $\mathbf{c} = \langle \overline{\nu}_\ell(\eta(z_i^\ell))\rangle_{\mathrm{inter}(j,t)} \in \mathsf{Tuples}^j_{I_\ell}(\overline{\nu}(z_i^\ell))$. Hence $I_0 \cap I_\ell \neq \emptyset$, which contradicts the fact that the composition $\gamma_0 \bullet \gamma_\ell$ is defined. Hence, the rule:

$$\overline{\mathsf{A}}_\iota(x_1,\ldots,x_{\#(\mathsf{A})},\eta(x_1),\ldots,\eta(x_{\#(\mathsf{A})})) \leftarrow$$
$$\exists y_1 \ldots \exists y_m \exists \eta(y_1) \ldots \exists \eta(y_m) \,.\, \overline{\psi} * \pi * \mathbin{\text{\Large$*$}}_{\ell=1}^{h} \overline{\mathsf{B}}^\ell_{\iota^\ell}(z^\ell_1,\ldots,z^\ell_{\#(\mathsf{B}^\ell)},\eta(z^\ell_1),\ldots,\eta(z^\ell_{\#(\mathsf{B}^\ell)}))$$

is well-formed and thus belongs to $\overline{\Delta}$. To obtain $\mathsf{h} \Vdash^{\overline{\nu}}_{\overline{\Delta}} \overline{\mathsf{A}}_\iota(x_1,\ldots,x_{\#\mathsf{A}},\eta(x_1),\ldots,\eta(x_{\#(\mathsf{A})}))$, by the definition of:

$$\overline{\psi} \stackrel{\mathsf{def}}{=} x_1 \mapsto \eta(x_1) * \mathbin{\text{\Large$*$}}_{x \in \mathrm{fv}(\psi)} \mathsf{CompStates}_\psi(x) * \mathbin{\text{\Large$*$}}_{i=1}^{\#\mathsf{A}} \mathsf{InterAtoms}_\psi(x_i)$$

it is sufficient to prove the following points:

- $\mathsf{h}_0 \Vdash^{\overline{\nu}} x_1 \mapsto \eta(x_1)$: by the definition $\overline{\nu}$, we have $\overline{\nu}(\eta(x_1)) = \mathsf{h}(\nu(x_1))$, hence it is sufficient to prove that $\mathrm{dom}(\mathsf{h}_0) = \{\nu(x_1)\}$. "$\subseteq$" Let $c \in \mathrm{dom}(\mathsf{h}_0)$ be a component. By the definition of $\mathsf{h}_0 = \mathsf{h} \setminus \bigcup_{\ell=1}^h \mathsf{h}_\ell$, we have $\mathrm{dom}(\mathsf{h}_0) = \mathrm{dom}(\mathsf{h}) \setminus \bigcup_{\ell=1}^h \mathrm{dom}(\mathsf{h}_\ell) = \mathrm{nodes}(\gamma) \setminus \bigcup_{\ell=1}^h \mathrm{nodes}(\gamma_\ell) = \mathrm{nodes}(\gamma_0)$, because $\mathsf{h} \in \mathbb{G}(\gamma)$ and $\mathsf{h}_\ell \in \mathbb{G}(\gamma_\ell)$, for all $\ell \in [1,h]$. Since $\gamma_0 \models^{\nu'}_\Delta \psi$, we have $c = \nu'(x)$, for some $x \in \mathrm{fv}(\psi)$. Suppose, for a contradiction, that x and x_1 are not the same variable, then $x \in \{z^\ell_1,\ldots,z^\ell_{\#\mathsf{B}_\ell}\}$, for some $\ell \in [1,h]$, because Δ is progressing (Def. 12). By Lemma 12, we obtain $c \in \mathrm{nodes}(\gamma_\ell)$, contradiction. Then $c = \nu'(x_1) = \nu(x_1)$. "$\supseteq$" Because Δ is progressing, $\psi = [x_1] * \varphi$, hence $\nu(x_1) = \nu'(x_1) \in \mathrm{nodes}(\gamma_0) = \mathrm{dom}(\mathsf{h}_0)$, because $\gamma_0 \models^{\nu'} \psi$.
- $\emptyset \Vdash^{\overline{\nu}} \mathsf{CompStates}_\psi(x)$, for each $x \in \mathrm{fv}(\psi)$: by the definition of $\overline{\nu}$, $\mathsf{h} \in \mathbb{G}(\gamma)$ and points 1 and 3 of Def. 14.
- $\emptyset \Vdash^{\overline{\nu}} \mathsf{InterAtoms}_\psi(x_i)$, for each $i \in [1,\#\mathsf{A}]$: by the definition of $\overline{\nu}$, $\mathsf{h} \in \mathbb{G}(\gamma)$, definition of $\iota(i,j)$, for all $i \in [1,\#\mathsf{A}]$ and $j \in [1,M]$ and point 2 of Def. 14.
- $\emptyset \Vdash^{\overline{\nu}} \pi$: because $(\emptyset,\emptyset,\rho) \models^{\nu'} \pi$ and $\overline{\nu}$ agrees with ν' over $\mathrm{fv}(\pi)$. □

Lemma 15. *Let Δ be a progressing SID and $\mathsf{A} \in \mathrm{def}(\Delta)$ be a predicate, such that $\mathsf{h} \Vdash^{\overline{\nu}}_{\overline{\Delta}} \overline{\mathsf{A}}_\iota(x_1,\ldots,x_{\#(\mathsf{A})},\eta(x_1),\ldots,\eta(x_{\#(\mathsf{A})}))$, for a map $\iota : [1,\#(\mathsf{A})] \times [1,m] \to 2^{[0,\mathfrak{B}-1]}$ and a store $\overline{\nu}$. Then, the following hold:*

1. *$\overline{\nu}(x_i) \in \mathrm{dom}(\mathsf{h})$ and $\mathsf{h}(\overline{\nu}(x_i)) = \overline{\nu}(\eta(x_i))$, for all $i \in [1,\#\mathsf{A}]$,*
2. *there exists a configuration γ, such that $\mathsf{h} \in \mathbb{G}(\gamma)$ and $\gamma \models^{\overline{\nu}}_{\overline{\Delta}} \mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}})$.*

Proof. By fixpoint induction on the definition of $\mathsf{h} \Vdash^{\overline{\nu}}_{\overline{\Delta}} \overline{\mathsf{A}}_\iota(x_1,\ldots,x_{\#(\mathsf{A})},\eta(x_1),\ldots,\eta(x_{\#(\mathsf{A})}))$. Consider the following well-formed rule from $\overline{\Delta}$:

$$\overline{\mathsf{A}}_\iota(x_1,\ldots,x_{\#(\mathsf{A})},\eta(x_1),\ldots,\eta(x_{\#(\mathsf{A})})) \leftarrow$$
$$\exists y_1 \ldots \exists y_m \exists \eta(y_1) \ldots \exists \eta(y_m) \,.\, \overline{\psi} * \pi * \mathbin{\text{\Large$*$}}_{\ell=1}^{h} \overline{\mathsf{B}}^\ell_{\iota^\ell}(z^\ell_1,\ldots,z^\ell_{\#(\mathsf{B}^\ell)},\eta(z^\ell_1),\ldots,\eta(z^\ell_{\#(\mathsf{B}^\ell)}))$$

such that $\mathsf{h} \Vdash^{\overline{\nu}'}_{\Delta} \overline{\psi} * \pi * \circledast_{\ell=1}^{h} \overline{\mathsf{B}}^{\ell}_{\iota^{\ell}}(z^{\ell}_1, \ldots, z^{\ell}_{\#(\mathsf{B}^{\ell})}, \eta(z^{\ell}_1), \ldots, \eta(z^{\ell}_{\#(\mathsf{B}^{\ell})}))$, where $\overline{\nu}'$ is a store that agrees with ν over $x_1, \ldots, x_{\#(\mathsf{A})}$ and $\eta(x_1), \ldots, \eta(x_{\#(\mathsf{A})})$.

(1) If $i = 1$ then $x_1 \mapsto \eta(x_1)$ is a subformula of $\overline{\psi}$, thus $\overline{\nu}(x_1) = \overline{\nu}'(x_1) \in \mathrm{dom}(\mathsf{h})$ and $\mathsf{h}(\overline{\nu}(x_1)) = \overline{\nu}'(\eta(x_1)) = \overline{\nu}(\eta(x_1))$. Otherwise, because Δ is progressing, $x_i \in \{z^{\ell}_1, \ldots, z^{\ell}_{\#\mathsf{B}_{\ell}}\}$, for some $\ell \in [1,h]$ and point (1) follows from the inductive hypothesis.

(2) There exist heaps $\mathsf{h}_0, \ldots, \mathsf{h}_h$, such that the following hold:

- $\mathrm{dom}(\mathsf{h}_i) \cap \mathrm{dom}(\mathsf{h}_j) = \emptyset$, for all $0 \leq i < j \leq h$ and $\mathsf{h} = \mathsf{h}_0 \cup \ldots \cup \mathsf{h}_h$,
- $\mathsf{h}_0 \Vdash^{\overline{\nu}'} \overline{\psi} * \pi$,
- $\mathsf{h}_{\ell} \Vdash^{\overline{\nu}'}_{\Delta} \overline{\mathsf{B}}^{\ell}_{\iota^{\ell}}(y^{\ell}_1, \ldots, y^{\ell}_{\#(\mathsf{B}^{\ell})}, \eta(y^{\ell}_1), \ldots, \eta(y^{\ell}_{\#(\mathsf{B}^{\ell})}))$, for all $\ell \in [1,h]$.

By the inductive hypothesis, there exist configurations $\gamma_1 = (\mathcal{C}_1, I_1, \rho_1), \ldots, \gamma_h = (\mathcal{C}_h, I_h, \rho_h)$, such that $\mathsf{h}_{\ell} \in \mathbb{G}(\gamma_{\ell})$ and $\gamma_{\ell} \models^{\overline{\nu}}_{\Delta} \mathsf{B}_{\ell}(z^{\ell}_1, \ldots, z^{\ell}_{\#(\mathsf{B}_{\ell})})$, for all $\ell \in [1,h]$. We define the configuration $\gamma_0 = (\mathcal{C}_0, I_0, \rho_0)$, as follows:

- $\mathcal{C}_0 \stackrel{\mathsf{def}}{=} \{\overline{\nu}(x_1)\}$,
- $I_0 \stackrel{\mathsf{def}}{=} \{(\overline{\nu}'(z_1), p_1, \ldots, \nu'(z_n), p_n) \mid \langle z_1.p_1, \ldots, z_n.p_n \rangle \text{ occurs in } \psi\}$,
- $\rho_0(\overline{\nu}'(z)) \stackrel{\mathsf{def}}{=} q_k$ if and only if $z@q_k$ occurs in ψ, otherwise ρ_0 is arbitrary.

Moreover, we define the state map ρ as $\rho(c) \stackrel{\mathsf{def}}{=} \rho_{\ell}(c)$ if $c \in \mathrm{dom}(\mathsf{h}_{\ell})$, for all $\ell \in [0,h]$ and $\rho(c)$ is arbitrary, for $c \notin \bigcup_{\ell=0}^{h} \mathrm{dom}(\mathsf{h}_{\ell})$. Since $\mathrm{dom}(\mathsf{h}_0), \ldots, \mathrm{dom}(\mathsf{h}_h)$ are pairwise disjoint, ρ is properly defined. First, we prove that the composition $(\mathcal{C}_0, I_0, \rho) \bullet \ldots \bullet (\mathcal{C}_h, I_h, \rho)$ is defined, namely that, for all $0 \leq i < j \leq h$, we have:

- $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$: If $i = 0$ then either $\mathcal{C}_0 = \emptyset$, in which case we are done, or $\mathcal{C}_0 = \{\overline{\nu}(x_1)\} = \{\overline{\nu}'(x_1)\}$. By the definition of $\overline{\psi} = x_1 \mapsto \eta(x_1) * \varphi$ and $\mathsf{h}_0 \Vdash^{\overline{\nu}'} \overline{\psi}$, we obtain $\overline{\nu}'(x_1) \in \mathrm{dom}(\mathsf{h}_0)$. Since $\mathrm{dom}(\mathsf{h}_0) \cap \mathrm{dom}(\mathsf{h}_j) = \emptyset$, we obtain $\overline{\nu}(x_1) \notin \mathrm{dom}(\mathsf{h}_j)$. Since $\mathsf{h}_j \in \mathbb{G}(\gamma_j)$, we obtain $\mathcal{C}_j \subseteq \mathrm{nodes}(\gamma_j) = \mathrm{dom}(\mathsf{h}_j)$, thus $\mathcal{C}_i \cap \bullet_j = \emptyset$. Else $i > 0$ and, since $\mathsf{h}_i \in \mathbb{G}(\gamma_i)$ and $\mathsf{h}_j \in \mathbb{G}(\gamma_j)$, we obtain $\mathcal{C}_i \subseteq \mathrm{nodes}(\gamma_i) = \mathrm{dom}(\mathsf{h}_i)$ and $\mathcal{C}_j \subseteq \mathrm{nodes}(\gamma_j) = \mathrm{dom}(\mathsf{h}_j)$. But $\mathrm{dom}(\mathsf{h}_i) \cap \mathrm{dom}(\mathsf{h}_j) = \emptyset$, leading to $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$.
- $I_i \cap I_j = \emptyset$: If $i = 0$, by the definition of I_0, each interaction from I_0 is of the form $(\overline{\nu}'(z_1), p_1, \ldots, \overline{\nu}'(z_n), p_n)$, such that $\langle z_1.p_1, \ldots, z_n.p_n \rangle$ is an interaction atom occuring in ψ. Since, moreover, Δ is progressing, we have $x_1 \in \{z_1, \ldots, z_n\}$, hence $\overline{\nu}(x_1) \in \{\overline{\nu}'(z_1), \ldots, \overline{\nu}'(z_n)\}$. Let $(c_1, p_1, \ldots, c_n, p_n) \in I_j$ be an interaction. Since $\mathsf{h}_j \in \mathbb{G}(\gamma_j)$, we obtain $\{c_1, \ldots, c_n\} \subseteq \mathrm{nodes}(\gamma_j) = \mathrm{dom}(\mathsf{h}_j)$, hence $\{c_1, \ldots, c_n\} \cap \mathrm{dom}(\mathsf{h}_0) = \{c_1, \ldots, c_n\} \cap \{\overline{\nu}(x_1)\} = \emptyset$, leading to $I_i \cap I_j = \emptyset$, because the choices of $(\overline{\nu}'(z_1), p_1, \ldots, \overline{\nu}'(z_n), p_n) \in I_i$ and $(c_1, p_1, \ldots, c_n, p_n) \in I_j$ are arbitrary. Else, $i > 0$ and let $(c^i_1, p_1, \ldots, c^i_n, p_n) \in I_i$, $(c^j_1, p_1, \ldots, c^j_n, p_n) \in I_j$ be two interactions of the same type. Since $\mathsf{h}_i \in \mathbb{G}(\gamma_i)$ and $\mathsf{h}_j \in \mathbb{G}(\gamma_j)$, we have $\{c^i_1, \ldots, c^i_n\} \subseteq \mathrm{nodes}(\gamma_i) = \mathrm{dom}(\mathsf{h}_i)$ and $\{c^j_1, \ldots, c^j_n\} \subseteq \mathrm{nodes}(\gamma_j) = \mathrm{dom}(\mathsf{h}_j)$, respectively. Since $\mathrm{dom}(\mathsf{h}_i) \cap \mathrm{dom}(\mathsf{h}_j) = \emptyset$, we obtain $\{c^i_1, \ldots, c^i_n\} \cap \{c^j_1, \ldots, c^j_n\} = \emptyset$. Since the choices of $(c^i_1, p_1, \ldots, c^i_n, p_n) \in I_i$ and $(c^j_1, p_1, \ldots, c^j_n, p_n) \in I_j$ are arbitrary, we obtain $I_i \cap I_j = \emptyset$.

Consequently, we define $\gamma \stackrel{\mathsf{def}}{=} (\mathcal{C}_0, I_0, \rho) \bullet \ldots \bullet (\mathcal{C}_h, I_h, \rho)$ and conclude by proving the following points:

– $\mathsf{h} = \mathbb{G}(\gamma)$: we prove that $\mathrm{dom}(\mathsf{h}) = \bigcup_{\ell=0}^{h} \mathrm{dom}(\mathsf{h}_\ell) = \mathrm{nodes}(\gamma)$, as required by Def. 14. The conditions (1-3) for $c_0 = \overline{\nu}(x_1)$ are by the definition of $\overline{\psi}$; for $c_0 \in \mathrm{dom}(\mathsf{h}_\ell)$ these conditions follow from $\mathsf{h}_\ell \in \mathbb{G}(\gamma_\ell)$. "$\subseteq$" Let $c \in \mathrm{dom}(\mathsf{h})$ be a component. If $c \in \mathrm{dom}(\mathsf{h}_0) = \{\overline{\nu}x_1\}$, then $c \in \mathcal{C}_0 \subseteq \mathrm{nodes}(\gamma_0) \subseteq \mathrm{nodes}(\gamma)$. Else $c \in \mathrm{dom}(\mathsf{h}_\ell)$, for some $\ell \in [1,h]$, then $c \in \mathrm{nodes}(\gamma_\ell) \subseteq \mathrm{nodes}(\gamma)$, because $\mathsf{h}_\ell \in \mathbb{G}(\gamma_\ell)$. "$\supseteq$" Let $c \in \mathrm{nodes}(\gamma) = \bigcup_{\ell=0}^{h} \mathrm{nodes}(\gamma_\ell)$ be a component. If $c \in \mathrm{nodes}(\gamma_0)$ then either $c \in \mathcal{C}_0$ or c occurs in some interaction from I_0. If $c \in \mathcal{C}_0$ then $c = \overline{\nu}(x_1) \in \mathrm{dom}(\mathsf{h}_0) \subseteq \mathrm{dom}(\mathsf{h})$, by the definition of $\mathcal{C}_0$. Else there exists an interaction $(c_1, p_1, \ldots, c_n, p_n) \in I_0$, such that $c \in \{c_1, \ldots, c_n\}$. In this case $c = \overline{\nu}'(z)$, for some variable z that occurs in an interaction atom from ψ. Since Δ is progressing, $z \in \{z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell\}$, for some $\ell \in [1,h]$. Because $\gamma_\ell \models_\Delta^{\overline{\nu}'} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$, we obtain $c = \overline{\nu}'(z) \in \mathrm{nodes}(\gamma_\ell)$, by Lemma 12, and $c \in \mathrm{dom}(\mathsf{h}_\ell) \subseteq \mathrm{dom}(\mathsf{h})$, because $\mathsf{h}_\ell \in \mathbb{G}(\gamma_\ell)$. If $c \in \mathrm{nodes}(\gamma_\ell)$, for some $\ell \in [1,h]$, we have $c \in \mathrm{dom}(\mathsf{h}_\ell) \subseteq \mathrm{dom}(\mathsf{h})$, because $\mathsf{h}_\ell \in \mathbb{G}(\gamma_\ell)$.
– $\gamma \models_\Delta^{\overline{\nu}} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$: Let the stem of the above rule from $\overline{\Delta}$ be:

$$\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_m \,.\, \psi * \pi * \mathbin{\ast\!\!\!\ast}_{\ell=1}^{h} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$$

Since $(\mathcal{C}_\ell, I_\ell, \rho_\ell) \models_\Delta^{\overline{\nu}'} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$ and ρ agrees with ρ_ℓ over $\mathrm{nodes}(\gamma_\ell)$, it follows that $(\mathcal{C}_\ell, I_\ell, \rho) \models_\Delta^{\overline{\nu}'} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$, for all $\ell \in [1,h]$. Moreover, $(\mathcal{C}_0, I_0, \rho) \models^{\overline{\nu}'} \psi$ by definition, and $(\emptyset, \emptyset, \rho) \models^{\overline{\nu}'} \pi$, because $\emptyset \Vdash^{\overline{\nu}'} \pi$. Altogether, we obtain $\gamma \models_\Delta^{\overline{\nu}'} \psi * \pi * \mathbin{\ast\!\!\!\ast}_{\ell=1}^{h} \mathsf{B}_\ell(z_1^\ell, \ldots, z_{\#\mathsf{B}_\ell}^\ell)$, leading to $\gamma \models_\Delta^{\overline{\nu}} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$. □

We state below the main result of this section on the complexity of the entailment problem. The upper bounds follow from a many-one reduction of $\mathsf{Entl}[\Delta, \mathsf{A}, \mathsf{B}]$ to the SL entailment $\overline{\mathsf{A}}_\iota(x_1, \ldots, x_{\#\mathsf{A}}, \eta(x_1), \ldots, \eta(x_{\#\mathsf{A}})) \Vdash_{\overline{\Delta}} \exists x_{\#\mathsf{B}+1} \ldots \exists x_{\#\mathsf{B}} \exists \eta(x_{\#\mathsf{B}+1}) \ldots \exists \eta(x_{\#\mathsf{B}})$. $\overline{\mathsf{B}}_{\iota'}(x_1, \ldots, x_{\#\mathsf{B}}, \eta(x_1), \ldots, \eta(x_{\#\mathsf{B}}))$, in combination with the upper bound provided by Theorem 5, for SL entailments. If $k < \infty$, the complexity is tight for CL, whereas gaps occur for $k = \infty, \ell < \infty$ and $k = \infty, \ell = \infty$, due to the cut-off on the degree bound (Prop. 1), which impacts the size of $\overline{\Delta}$ and time needed to generate it from Δ.

Theorem 6. *If Δ is progressing, connected and e-restricted and, moreover,* $\mathsf{Bnd}[\Delta, \mathsf{A}]$ *has a positive answer,* $\mathsf{Entl}^{k,\ell}[\Delta, \mathsf{A}, \mathsf{B}]$ *is in* 2EXP, $\mathsf{Entl}^{\infty,\ell}[\Delta, \mathsf{A}, \mathsf{B}]$ *is in* 3EXP $\cap$ 2EXP*-hard, and* $\mathsf{Entl}[\Delta, \mathsf{A}, \mathsf{B}]$ *is in* 4EXP $\cap$ 2EXP*-hard.*

Proof. The proof consists of three parts. (1) We reduce $\mathsf{Entl}[\Delta, \mathsf{A}, \mathsf{B}]$ to an equivalent SL entailment problem, for a progressing, connected and e-restricted SID. (2) This reduction provides upper bounds for $\mathsf{Entl}^{k,\ell}[\Delta, \mathsf{A}, \mathsf{B}]$, in the cases $k, \ell < \infty$, $k = \infty, \ell < \infty$ and $k = \ell = \infty$, respectively. (3) We give a lower bound for $\mathsf{Entl}[\Delta, \mathsf{A}, \mathsf{B}]$, by reduction from SL entailment.

(1) We prove that, for each map $\iota : [1, \#\mathsf{A}] \times [1, M] \to 2^{[0, 2\mathfrak{B}-1]}$ there exists a map $\iota' : [1, \#\mathsf{B}] \times [1, M] \to 2^{[0, 2\mathfrak{B}-1]}$, such that:

$$\begin{array}{rl} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \models_\Delta & \exists x_{\#\mathsf{B}+1} \ldots \exists x_{\#\mathsf{B}} \,.\, \mathsf{B}(x_1, \ldots, x_{\#\mathsf{B}}) \iff \\ \overline{\mathsf{A}}_\iota(x_1, \ldots, x_{\#\mathsf{A}}, \eta(x_1), \ldots, \eta(x_{\#\mathsf{A}})) \Vdash_{\overline{\Delta}} & \exists x_{\#\mathsf{B}+1} \ldots \exists x_{\#\mathsf{B}} \exists \eta(x_{\#\mathsf{B}+1}) \ldots \exists \eta(x_{\#\mathsf{B}}) \,. \\ & \overline{\mathsf{B}}_{\iota'}(x_1, \ldots, x_{\#\mathsf{B}}, \eta(x_1), \ldots, \eta(x_{\#\mathsf{B}})) \end{array}$$

"$\Rightarrow$" Let h be a heap and $\overline{\nu}$ be a store, such that $\mathsf{h} \Vdash^{\overline{\nu}}_{\overline{\Delta}} \overline{\mathsf{A}}_{\iota}(x_1, \ldots, x_{\#\mathsf{A}}, \eta(x_1), \ldots, \eta(x_{\#\mathsf{A}}))$. By Lemma 15, we have $\mathsf{h}(\overline{\nu}(x_i)) = \overline{\nu}(\eta(x_i))$, for all $i \in [1, \#\mathsf{A}]$ and, moreover, there exists a configuration γ, such that $\mathsf{h} \in \mathbb{G}(\gamma)$ and $\gamma \models^{\overline{\nu}}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$. By the hypothesis, we obtain $\gamma \models^{\overline{\nu}}_{\Delta} \exists x_{\#\mathsf{A}+1} \ldots \exists x_{\#\mathsf{B}} \,.\, \mathsf{B}(x_1, \ldots, x_{\#\mathsf{B}})$, hence there exists a store $\overline{\nu}'$, that agrees with $\overline{\nu}$ over $x_1, \ldots, x_{\#\mathsf{A}}$, such that $\gamma \models^{\overline{\nu}'}_{\Delta} \mathsf{B}(x_1, \ldots, x_{\#\mathsf{B}})$. By Lemma 14, there exists a store $\overline{\nu}''$ that agrees with $\overline{\nu}'$ over $x_1, \ldots, x_{\#\mathsf{B}}$, such that $\mathsf{h}(\overline{\nu}''(x_i)) = \overline{\nu}''(\eta(x_i))$, for all $i \in [1, \#\mathsf{B}]$ and $\mathsf{h} \Vdash^{\overline{\nu}''}_{\overline{\Delta}} \overline{\mathsf{B}}_{\iota'}(x_1, \ldots, x_{\#\mathsf{B}}, \eta(x_1), \ldots, \eta(x_{\#\mathsf{B}}))$, for some map $\iota' : [1, \#\mathsf{B}] \times [1, M] \to 2^{[0, \mathfrak{B}-1]}$, because $\mathsf{h} \in \mathbb{G}(\gamma)$. Hence, $\overline{\nu}''$ agrees with $\overline{\nu}$ over $x_1, \ldots, x_{\#\mathsf{A}}, \eta(x_1), \ldots, \eta(\#\mathsf{A})$, thus we obtain $\mathsf{h} \Vdash^{\overline{\nu}}_{\overline{\Delta}} \exists x_{\#\mathsf{A}+1} \ldots \exists x_{\#\mathsf{B}} \,.\, \overline{\mathsf{B}}_{\iota'}(x_1, \ldots, x_{\#\mathsf{B}}, \eta(x_1), \ldots, \eta(x_{\#\mathsf{B}}))$.

"$\Leftarrow$" Let γ be a configuration, ν be a store such that $\gamma \models^{\nu}_{\Delta} \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}})$ and $\mathsf{h} \in \mathbb{G}(\gamma)$ be a heap. Cleary, such a heap exists, for any given configuration, by Def. 14. By Lemma 14, there exists a map $\iota : [1, \#\mathsf{A}] \times [1, M] \to 2^{[0, \mathfrak{B}-1]}$ and a store $\overline{\nu}$, that agrees with ν over $x_1, \ldots, x_{\#\mathsf{A}}$, such that $\overline{\nu}(\eta(x_i)) = \mathsf{h}(\nu(x_i))$, for all $i \in [1, \#\mathsf{A}]$ and $\mathsf{h} \Vdash^{\overline{\nu}}_{\overline{\Delta}} \overline{\mathsf{A}}_{\iota}(x_1, \ldots, x_{\#\mathsf{A}}, \eta(x_1), \ldots, \eta(x_{\#\mathsf{A}}))$. By the hypothesis, we have $\mathsf{h} \Vdash^{\overline{\nu}'}_{\overline{\Delta}} \overline{\mathsf{B}}_{\iota'}(x_1, \ldots, x_{\#\mathsf{B}}, \eta(x_1), \ldots, \eta(x_{\#\mathsf{B}}))$, for some map $\iota' : [1, \#\mathsf{B}] \times [1, M] \to 2^{[0, \mathfrak{B}-1]}$ and a store $\overline{\nu}'$ that agrees with $\overline{\nu}$ over $x_1, \ldots, x_{\#\mathsf{A}}$ and $\eta(x_1), \ldots, \eta(x_{\#\mathsf{A}})$. By Lemma 15, we have $\overline{\nu}'(\eta(x_i)) = \mathsf{h}(\overline{\nu}(x_i)) = \overline{\nu}(\eta(x_i))$, for all $i \in [1, \#\mathsf{A}]$ and there exists a configuration γ', such that $\mathsf{h} \in \mathbb{G}(\gamma')$ and $\gamma' \models^{\overline{\nu}'}_{\Delta} \mathsf{B}(x_1, \ldots, x_{\#\mathsf{B}})$. Since $\mathsf{h} \in \mathbb{G}(\gamma) \cap \mathbb{G}(\gamma')$, by Def. 14, we obtain $\gamma = \gamma'$, hence $\gamma \models^{\overline{\nu}'}_{\Delta} \mathsf{B}(x_1, \ldots, x_{\#\mathsf{B}})$. Since, moreover $\overline{\nu}'$, $\overline{\nu}$ and ν all agree over $x_1, \ldots, x_{\#\mathsf{A}}$, we obtain $\gamma \models^{\nu}_{\Delta} \exists x_{\#\mathsf{A}+1} \ldots \exists x_{\#\mathsf{B}} \,.\, \mathsf{B}(x_1, \ldots, x_{\#\mathsf{B}})$.

Since Δ is progressing and connected, $\overline{\Delta}$ is progressing and connected as well. Moreover, $\overline{\Delta}$ is e-restricted, because Δ is e-restricted and the construction of $\overline{\Delta}$ only introduces equalities, not disequalities.

(2) The upper bound relies on the result of [19, Theorem 32], that gives a $2^{2^{\mathsf{poly}(\mathrm{width}(\overline{\Delta}) \cdot \log \mathrm{size}(\overline{\Delta}))}}$ upper bound for SL entailments. Note that the number of variables in each rule from $\overline{\Delta}$ is the number of variables in its stem rule multiplied by $\mathfrak{K}+1$, hence $\mathrm{width}(\overline{\Delta}) \leq \mathrm{width}(\Delta) \cdot (\mathfrak{K}+1) = \mathrm{width}(\Delta) \cdot \mathfrak{B} \cdot \mathrm{intersize}(\Delta) \cdot 2^{O(\mathrm{intersize}(\Delta))}$, because $\mathfrak{K} = \mathrm{pos}(0, M+1, N) = 1 + \mathfrak{B} \cdot \sum_{\ell=1}^{M} |\tau_\ell| + N = \mathfrak{B} \cdot \mathrm{intersize}(\Delta) \cdot 2^{O(\mathrm{intersize}(\Delta))}$. The time needed to build $\overline{\Delta}$ and its size are bounded as follows:

$$
\begin{aligned}
\mathrm{size}(\overline{\Delta}) &\leq \|\overline{\Delta}\| \cdot \mathrm{width}(\overline{\Delta}), \text{ since there are } 2^{\mathfrak{B} \cdot M \cdot \mathrm{arity}(\Delta)} \text{ maps } \iota : [1, \#\mathsf{A}] \times [1, M] \to 2^{[0, \mathfrak{B}-1]} \\
&\leq 2^{\mathfrak{B} \cdot M \cdot \mathrm{arity}(\Delta)} \cdot \|\Delta\| \cdot \mathrm{width}(\overline{\Delta}) \\
&= 2^{\mathfrak{B} \cdot 2^{\mathrm{intersize}(\Delta)} \cdot \mathrm{arity}(\Delta)} \cdot \|\Delta\| \cdot \mathfrak{B} \cdot \mathrm{intersize}(\Delta) \cdot 2^{O(\mathrm{intersize}(\Delta))} \\
&= \mathrm{size}(\Delta) \cdot 2^{poly(\mathfrak{B} \cdot 2^{\mathrm{intersize}(\Delta)} \cdot \mathrm{arity}(\Delta))}
\end{aligned}
$$

By Prop. 1, we consider the following cases:

- if $k, \ell < \infty$ then $\mathfrak{B} = poly(\mathrm{size}(\Delta))$, thus $\mathrm{size}(\overline{\Delta}) = 2^{2^{O(\mathrm{size}(\mathrm{size}(\Delta)))}}$
- if $k = \infty$ and $\ell < \infty$ then $\mathfrak{B} = 2^{poly(\mathrm{size}(\Delta))}$, thus $\mathrm{size}(\overline{\Delta}) = 2^{2^{2^{O(\mathrm{size}(\mathrm{size}(\Delta)))}}}$
- if $k = \infty$ and $\ell = \infty$ then $\mathfrak{B} = 2^{2^{poly(\mathrm{size}(\Delta))}}$, thus $\mathrm{size}(\overline{\Delta}) = 2^{2^{2^{O(\mathrm{size}(\mathrm{size}(\Delta)))}}}$

(3) The 2EXP-hard lower bound for $\mathsf{Entl}^{\infty,\ell}[\Delta,\mathsf{A},\mathsf{B}]$ and $\mathsf{Entl}[\Delta,\mathsf{A},\mathsf{B}]$ is obtained by reduction from the SL entailment problem $\overline{\mathsf{A}}(x_1,\ldots,x_k) \Vdash_{\overline{\Delta}} \overline{\mathsf{B}}(x_1,\ldots,x_k)$, where $\overline{\Delta}$ is a progressing and connected SID, with no disequalities [18, Theorem 18]. Note that the maximum arity of Δ cannot be bounded to a constant, in order to obtain 2EXP-hardness of the SL entailment problem, hence the lower bound does not apply to $\mathsf{Entl}^{k,\ell}[\Delta,\mathsf{A},\mathsf{B}]$. The idea of the reduction is to encode each SL atomic proposition of the form $x \mapsto (y_1,\ldots,y_{\mathfrak{K}})$ by the formula $[x] * \langle x.p_0,\ldots,y_{\mathfrak{K}}.p_{\mathfrak{K}}\rangle$. Then each model h of a SL predicate atom $\overline{\mathsf{A}}(x_1,\ldots,x_{\#(\overline{\mathsf{A}})})$ is represented by a configuration $\gamma = (C, I, \rho)$, such that $C = \mathrm{dom}(\mathsf{h})$ and $I = \{(c_0,p_0,\ldots,c_{\mathfrak{K}},p_{\mathfrak{K}}) \mid \mathsf{h}(c_0) = \langle c_1,\ldots,c_{\mathfrak{K}}\rangle\}$. Since $\overline{\Delta}$ is progressing and connected, the CL SID Δ, obtained from the reduction, is progressing and connected. Since, moreover, the reduction does not introduce disequalities, Δ is trivially e-restricted. Because the reduction takes polynomial time, we obtain a 2EXP-hard lower bound. □

7 Conclusions and Future Work

We study the satisfiability and entailment problems in a logic used to write proofs of correctness for dynamically reconfigurable distributed systems. The logic views the components and interactions from the network as resources and reasons also about the local states of the components. We reuse existing techniques for Separation Logic [40], showing that our configuration logic is more expressive than SL, fact which is confirmed by a number of complexity gaps. Closing up these gaps and finding tight complexity classes in the more general cases is considered for future work. In particular, we aim at lifting the boundedness assumption on the degree of the configurations that must be considered to check the validity of entailments.

References

1. E. Ahrens, M. Bozga, R. Iosif, and J. Katoen. Local reasoning about parameterized reconfigurable distributed systems. *CoRR*, abs/2107.05253, 2021.
2. F. Arbab. Reo: A channel-based coordination model for component composition. *Mathematical. Structures in Comp. Sci.*, 14(3):329–366, June 2004.
3. Y. Bar-Hillel, M. Perles, and E. Shamir. On formal properties of simple phrase structure grammars. *Sprachtypologie und Universalienforschung*, 14:143–172, 1961.
4. A. Basu, M. Bozga, and J. Sifakis. Modeling heterogeneous real-time components in BIP. In *Fourth IEEE International Conference on Software Engineering and Formal Methods (SEFM 2006)*, pages 3–12. IEEE Computer Society, 2006.
5. R. Bloem, S. Jacobs, A. Khalimov, I. Konnov, S. Rubin, H. Veith, and J. Widder. *Decidability of Parameterized Verification*. Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2015.
6. M. Bozga, R. Iosif, and J. Sifakis. Verification of component-based systems with recursive architectures. *CoRR*, abs/2112.08292, 2021.
7. J. Bradbury, J. Cordy, J. Dingel, and M. Wermelinger. A survey of self-management in dynamic software architecture specifications. In *Proceedings of the 1st ACM SIGSOFT workshop on Self-managed systems*, pages 28–33. ACM, 2004.
8. S. Brookes and P. W. O'Hearn. Concurrent separation logic. *ACM SIGLOG News*, 3(3):47–65, Aug. 2016.

9. J. Brotherston, C. Fuhs, J. A. N. Pérez, and N. Gorogiannis. A decision procedure for satisfiability in separation logic with inductive predicates. In *CSL-LICS*, pages 25:1–25:10. ACM, 2014.
10. A. Bucchiarone and J. P. Galeotti. Dynamic software architectures verification using dynalloy. *Electron. Commun. Eur. Assoc. Softw. Sci. Technol.*, 10, 2008.
11. A. Butting, R. Heim, O. Kautz, J. O. Ringert, B. Rumpe, and A. Wortmann. A classification of dynamic reconfiguration in component and connector architecture description. In *Proceedings of MODELS 2017 Satellite Event: Workshops (ModComp)*, volume 2019 of *CEUR Workshop Proceedings*, pages 10–16. CEUR-WS.org, 2017.
12. C. Calcagno, P. W. O'Hearn, and H. Yang. Local action and abstract separation logic. In *22nd IEEE Symposium on Logic in Computer Science (LICS 2007), 10-12 July 2007, Wroclaw, Poland, Proceedings*, pages 366–378. IEEE Computer Society, 2007.
13. E. Cavalcante, T. V. Batista, and F. Oquendo. Supporting dynamic software architectures: From architectural description to implementation. In L. Bass, P. Lago, and P. Kruchten, editors, *12th Working IEEE/IFIP Conference on Software Architecture, WICSA 2015*, pages 31–40. IEEE Computer Society, 2015.
14. D. Clarke. A basic logic for reasoning about connector reconfiguration. *Fundam. Inf.*, 82(4):361–390, Feb. 2008.
15. T. Dinsdale-Young, L. Birkedal, P. Gardner, M. Parkinson, and H. Yang. Views: Compositional reasoning for concurrent programs. *SIGPLAN Not.*, 48(1):287–300, Jan. 2013.
16. T. Dinsdale-Young, M. Dodds, P. Gardner, M. J. Parkinson, and V. Vafeiadis. Concurrent abstract predicates. In *ECOOP 2010 – Object-Oriented Programming*, pages 504–528, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
17. J. Dormoy, O. Kouchnarenko, and A. Lanoix. Using temporal logic for dynamic reconfigurations of components. In L. S. Barbosa and M. Lumpe, editors, *Formal Aspects of Component Software - 7th International Workshop, FACS 2010*, volume 6921 of *Lecture Notes in Computer Science*, pages 200–217. Springer, 2010.
18. M. Echenim, R. Iosif, and N. Peltier. Entailment checking in separation logic with inductive definitions is 2-exptime hard. In *LPAR 2020: 23rd International Conference on Logic for Programming, Artificial Intelligence and Reasoning, Alicante, Spain, May 22-27, 2020*, volume 73 of *EPiC Series in Computing*, pages 191–211. EasyChair, 2020.
19. M. Echenim, R. Iosif, and N. Peltier. Decidable Entailments in Separation Logic with Inductive Definitions: Beyond Establishment. In C. Baier and J. Goubault-Larrecq, editors, *29th EACSL Annual Conference on Computer Science Logic (CSL 2021)*, volume 183 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:18, Dagstuhl, Germany, 2021. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
20. M. Echenim, R. Iosif, and N. Peltier. Unifying decidable entailments in separation logic with inductive definitions. In *28th International Conference on Automated Deduction (CADE'21), Proceedings, to appear*, 2021.
21. R. El-Ballouli, S. Bensalem, M. Bozga, and J. Sifakis. Programming dynamic reconfigurable systems. *International Journal on Software Tools for Technology Transfer*, January 2021.
22. A. El-Hokayem, M. Bozga, and J. Sifakis. A temporal configuration logic for dynamic reconfigurable systems. In C. Hung, J. Hong, A. Bechini, and E. Song, editors, *SAC '21: The 36th ACM/SIGAPP Symposium on Applied Computing, Virtual Event, Republic of Korea, March 22-26, 2021*, pages 1419–1428. ACM, 2021.
23. F. Farka, A. Nanevski, A. Banerjee, G. A. Delbianco, and I. Fábregas. On algebraic abstractions for concurrent separation logics. *Proc. ACM Program. Lang.*, 5(POPL), Jan. 2021.
24. X. Feng, R. Ferreira, and Z. Shao. On the relationship between concurrent separation logic and assume-guarantee reasoning. In *Programming Languages and Systems*, pages 173–188. Springer Berlin Heidelberg, 2007.

25. K. Foerster and S. Schmid. Survey of reconfigurable data center networks: Enablers, algorithms, complexity. *SIGACT News*, 50(2):62–79, 2019.
26. H. Gaifman. On local and non-local properties. *Studies in logic and the foundations of mathematics*, 107:105–135, 1982.
27. D. Hirsch, P. Inverardi, and U. Montanari. Graph grammars and constraint solving for software architecture styles. In *Proceedings of the Third International Workshop on Software Architecture*, ISAW '98, page 69–72, New York, NY, USA, 1998. Association for Computing Machinery.
28. C. Jansen, J. Katelaan, C. Matheja, T. Noll, and F. Zuleger. Unified reasoning about robustness properties of symbolic-heap separation logic. In *ESOP*, volume 10201 of *Lecture Notes in Computer Science*, pages 611–638. Springer, 2017.
29. C. B. Jones. *Developing methods for computer programs including a notion of interference*. PhD thesis, University of Oxford, UK, 1981.
30. I. V. Konnov, T. Kotek, Q. Wang, H. Veith, S. Bliudze, and J. Sifakis. Parameterized systems in BIP: design and model checking. In *27th International Conference on Concurrency Theory, CONCUR 2016*, volume 59 of *LIPIcs*, pages 30:1–30:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.
31. C. Krause, Z. Maraikar, A. Lazovik, and F. Arbab. Modeling dynamic reconfigurations in reo using high-level replacement systems. *Science of Computer Programming*, 76:23–36, 01 2011.
32. A. Lanoix, J. Dormoy, and O. Kouchnarenko. Combining proof and model-checking to validate reconfigurable architectures. *Electron. Notes Theor. Comput. Sci.*, 279(2):43–57, 2011.
33. D. Le Metayer. Describing software architecture styles using graph grammars. *IEEE Transactions on Software Engineering*, 24(7):521–533, 1998.
34. J. Magee and J. Kramer. Dynamic structure in software architectures. In *ACM SIGSOFT Software Engineering Notes*, volume 21(6), pages 3–14. ACM, 1996.
35. A. Mavridou, E. Baranov, S. Bliudze, and J. Sifakis. Configuration logics: Modeling architecture styles. *J. Log. Algebr. Meth. Program.*, 86(1):2–29, 2017.
36. M. Noormohammadpour and C. S. Raghavendra. Datacenter traffic control: Understanding techniques and tradeoffs. *IEEE Commun. Surv. Tutorials*, 20(2):1492–1525, 2018.
37. P. W. O'Hearn. Resources, concurrency, and local reasoning. *Theor. Comput. Sci.*, 375(1-3):271–307, 2007.
38. P. W. O'Hearn and D. J. Pym. The logic of bunched implications. *Bull. Symb. Log.*, 5(2):215–244, 1999.
39. S. Owicki and D. Gries. *An Axiomatic Proof Technique for Parallel Programs*, pages 130–152. Springer New York, New York, NY, 1978.
40. J. C. Reynolds. Separation logic: A logic for shared mutable data structures. In *17th IEEE Symposium on Logic in Computer Science (LICS 2002), 22-25 July 2002, Copenhagen, Denmark, Proceedings*, pages 55–74. IEEE Computer Society, 2002.
41. Z. Shtadler and O. Grumberg. Network grammars, communication behaviors and automatic verification. In J. Sifakis, editor, *Automatic Verification Methods for Finite State Systems, International Workshop*, volume 407 of *LNCS*, pages 151–165. Springer, 1989.
42. G. Taentzer, M. Goedicke, and T. Meyer. Dynamic change management by distributed graph transformation: Towards configurable distributed systems. In *International Workshop on Theory and Application of Graph Transformations*, pages 179–193. Springer, 1998.
43. V. Vafeiadis and M. Parkinson. A marriage of rely/guarantee and separation logic. In *CONCUR 2007 – Concurrency Theory*, pages 256–271. Springer Berlin Heidelberg, 2007.
44. M. Wermelinger. Towards a chemical model for software architecture reconfiguration. *IEE Proceedings-Software*, 145(5):130–136, 1998.

45. M. Wermelinger and J. L. Fiadeiro. A graph transformation approach to software architecture reconfiguration. *Sci. Comput. Program.*, 44(2):133–155, 2002.

A Proofs