Cyclic Proofs with Ordering Constraints

Sorin Stratulat

Cyclic Proofs with Ordering Constraints

Sorin Stratulat

LORIA, Department of Computer Science
Université de Lorraine
Metz, F-57000, FRANCE
sorin.stratulat@univ-lorraine.fr

Abstract. CLKID$^\omega$ is a sequent-based cyclic inference system able to reason on first-order logic with inductive definitions. The current approach for verifying the soundness of CLKID$^\omega$ proofs is based on expensive model-checking techniques leading to an explosion in the number of states.
We propose proof strategies that guarantee the soundness of a class of CLKID$^\omega$ proofs if some ordering and derivability constraints are satisfied. They are inspired from previous works about cyclic well-founded induction reasoning, known to provide effective sets of ordering constraints. A derivability constraint can be checked in linear time. Under certain conditions, one can build proofs that implicitly satisfy the ordering constraints.

1 Introduction

CLKID$^\omega$ [9] is the *de facto* standard sequent-based cyclic inference system for performing lazy induction reasoning on specifications based on first-order logic with inductive definitions ($\mathrm{FOL_{ID}}$). The CLKID$^\omega$ proofs are represented as finite derivation trees with nodes labelled by sequents. A particular feature is that cycles can be built by establishing connections between terminal and non-terminal nodes labelled with identical sequents. The soundness of CLKID$^\omega$ proofs is entailed from some global trace condition by using Infinite Descent induction arguments [20]. This condition requires that, for every infinite path in the cyclic derivation of a false sequent, all successive steps starting from some point are decreasing and certain steps occurring infinitely often are strictly decreasing w.r.t. some semantic ordering.

CLKID$^\omega$ has been implemented in the Cyclist prover [8]. Since the global trace condition is an ω-regular property, Cyclist can check it during the proof construction or *post hoc* as an inclusion between two Büchi automata by calling an external model checker. It turns out that the inclusion test may be costly. Indeed, for any proof P, the approach requires the construction of the automaton complementary to that accepting strings over infinite progressing traces in P, based on a complementation method for Büchi automata as described in [11]. The method ensures that, for every automaton with n states, the generated complementary automaton has at least $2^{O(n \log n)}$ states [12]. In case of failure

of the inclusion test, previous proof steps should be reconsidered, requiring that existing connections be broken, proof steps cancelled or different inference rules applied. Hence, it may happen that the test be executed several times during the proof construction. For the proofs of the toy examples from [8], the percentage of time taken by the soundness check include values from 0% to 44%.

Example 1.1. The 'P and Q' example [20] is specified in [8] using the following mutually dependent inductive predicates P and Q defined over naturals:

$$\Rightarrow P(0) \quad (1) \qquad \Rightarrow Q(x,0) \quad (3)$$
$$P(x) \wedge Q(x,s(x)) \Rightarrow P(s(x)) \quad (2) \qquad P(x) \wedge Q(x,y) \Rightarrow Q(x,s(y)) \quad (4)$$

where 0 and s are the usual constructor symbols for naturals. Let N define the set of naturals by the productions:

$$\Rightarrow N(0) \quad (5) \qquad N(x) \Rightarrow N(s(x)) \quad (6)$$

According to Table 1 from [8], CYCLIST can prove the sequent $N(x), N(y) \vdash Q(x,y)$ in about half a second, by building a proof tree with only 13 nodes. The validation process required 181 calls to the external model checker, among which 171 calls are failing. 31% of the time is spent on the soundness check.

On the other hand, a different approach based on ordering constraints has been proposed in [15,17] for performing cyclic well-founded induction to check inductive consequences of conditional specifications. The proofs generated by this approach are normalized to sets of tree derivations and represented as directed graphs (for short, digraphs) allowing some terminal nodes to be connected to root nodes. The minimal cycles resulting by following the arrows in the digraph are denoted as cyclic lists of paths leading a root to a terminal node in the same tree derivation. The ordering constraints for checking the proof soundness involve only comparisons between instances of root formulas. Their number is given by the number of paths from the minimal cycles and does not depend on the length of these paths. Cyclic well-founded induction proofs have been validated in [16] by certifying environments as Coq [19].

This approach is rather general and helps to define *reductive* inference systems [5,15], as those based on implicit induction [4]. They can build automatically proof derivations whose soundness is implicitly guaranteed by the proof method, hence no validation steps are required. Lacking the inconvenients presented by the on-the-fly/*post-hoc* soundness tests or backtracking steps, they allow for an effective proof generation and help to deal with industrial-size applications [3,14].

This paper, structured in four sections, presents an effective solution to validate the global trace condition for proofs generated with CLKID$^{\omega}_{N}$, a restricted version of CLKID$^{\omega}$ using ordering constraints, in the same line as for cyclic well-founded induction proofs. Section 2 is a quick presentation to the logical framework based on FOL$_{\mathrm{ID}}$ and an introduction to CLKID$^{\omega}_{N}$. In Section 3, we define

proof strategies that guarantee the global trace condition for CLKID$^\omega_N$ proofs satisfying a set of ordering and derivability constraints. An ordering constraint consists of a comparison between two sequents, defined as a multiset extension of an ordering $<_a$ over literals. It can be decided in polynomial time in the size of the sequents if the $<_a$-relations can also be decided in polynomial time. The derivability constraints can be checked in linear time provided that the 'history' of some atoms from the compared sequents is preserved. We took as running example the conjecture from Example 6 of [8], whose proof required the maximal time percentage for the soundness check. We provide a CLKID$^\omega_N$ proof of it and show that its soundness check needs only two ordering constraints. A link with the reductive reasoning techniques is established; as a proof of concept, we define proof strategies that can build a reductive proof of $N(x), N(y) \vdash Q(x, y)$ and show that the ordering constraints are implicitly satisfied. The conclusions and future work are given in the last section.

2 The logical framework

The logical setting relies on FOL_{ID} with equality, as presented, e.g., in [7,9].

Syntax. Let Σ be a (countable) language built on a finite alphabet of arity-fixed function symbols $\mathcal{F}$ and predicate symbols, and $\mathcal{V}$ an enumerable set of variables. Each predicate symbol is either *inductive* (i.e., defined by axioms as below) or *ordinary* (i.e., not inductive). Terms and formulas are defined as usual. $(t_1, \ldots, t_n)$ denotes a vector of terms and $P(t_1, \ldots, t_n)$ an *inductive atom*, where P is an inductive predicate symbol and $t_1, \ldots, t_n$ are terms. $\equiv$ represents the syntactic equality. A substitution σ is a finite non-empty set of mappings of distinct variables to terms $\bigcup_{i=1}^{n}\{x_i \mapsto t_i\}$, also denoted as $\{\overline{x} \mapsto \overline{t}\}$, where $\overline{x} \equiv (x_1, \ldots, x_n)$ and $\overline{t} \equiv (t_1, \ldots, t_n)$. $t[\sigma]$ denotes the *instance* of a term t built with the substitution σ; we also say that $t[\sigma]$ *matches* t. Similarly, we can apply substitutions and build instances for atoms, formulas and (multi)sets of formulas. $FV(S)$ denotes the set of free variables from the set of formulas S.

Each inductive predicate symbol P is defined by a finite inductive definition set of productions (axioms) consisting of implication formulas of the form

$$(\textstyle\bigwedge_{m=1}^{h} Q_m(\overline{u}_m) \wedge \bigwedge_{m=1}^{l} P_{i_m}(\overline{t}_m)) \Rightarrow P(\overline{t}), \tag{7}$$

where $h, l, i_1, \ldots, i_l$ are naturals and $Q_1, \ldots, Q_h$ (resp., $P_{i_1}, \ldots, P_{i_l}$) are ordinary (resp., inductive) predicate symbols. (7) is an *unconditional* production if $h = 0$ and $l = 0$. If not, (7) is a *conditional* production and $\bigwedge_{m=1}^{h} Q_m(\overline{u}_m) \wedge \bigwedge_{m=1}^{l} P_{i_m}(\overline{t}_m)$ is its *condition*. Φ denotes the set of productions defining each inductive predicate symbol.

Orderings. Let $(\mathcal{E}, \leq)$ be a non-empty poset. The strict part of the partial order $\leq$, referred to as *ordering*, is denoted by $<$. A binary relation R is *stable under substitutions* if whenever $s\,R\,t$ then $(s[\sigma])\,R\,(t[\sigma])$, for every substitution

σ and terms/formulas s and t. Given two finite multisets A and B of elements from $\mathcal{E}$, we say that $\ll$ is the *multiset extension* of $<$ and write $B \ll A$ if there are two finite multisets X and Y such that $B = (A - X) \uplus Y$, $X \neq \emptyset$ and $\forall y \in Y,\ \exists x \in X, y < x$ holds, where $\uplus$ (resp., $-$) is the union (resp., difference) on multisets. In practice, X (resp., Y) is A (resp., B) after having deleted pairwisely the common elements.

The CLKID$^\omega_N$ inference system. CLKID$^\omega_N$ consists of a finite set of inference rules that process sequents [10] of the form $\Gamma \vdash \Delta$, where Γ and Δ are finite multisets of first-order formulas and referred to as *antecedents* and *succedents*, respectively. An inference rule transforms a sequent, called *conclusion*, into a (potentially empty) multiset of sequents, called *premises*; they are separated by a horizontal line followed by the name of the rule. Most of the CLKID$^\omega_N$ inference rules transform one (principal) formula from the conclusion. In this case, it is explicitly represented in the sequent. A more detailed presentation of the sequent calculus can be found elsewhere, e.g., [13].

$$\frac{}{\Gamma \vdash \Delta}\ \Gamma \cap \Delta \neq \emptyset\ (Ax) \qquad \frac{\Gamma' \vdash \Delta'}{\Gamma \vdash \Delta}\ \Gamma' \subseteq \Gamma, \Delta' \subseteq \Delta\ (Wk) \qquad \frac{\Gamma \vdash F, \Delta \quad \Gamma \vdash G, \Delta}{\Gamma \vdash F \wedge G, \Delta}\ (\wedge R)$$

$$\frac{\Gamma \vdash F, \Delta}{\Gamma, \neg F \vdash \Delta}\ (\neg L) \qquad \frac{\Gamma, F \vdash \Delta}{\Gamma \vdash \neg F, \Delta}\ (\neg R) \qquad \frac{\Gamma, F \vdash \Delta \quad \Gamma, G \vdash \Delta}{\Gamma, F \vee G \vdash \Delta}\ (\vee L)$$

$$\frac{\Gamma \vdash F, G, \Delta}{\Gamma \vdash F \vee G, \Delta}\ (\vee R) \qquad \frac{\Gamma, F, G \vdash \Delta}{\Gamma, F \wedge G \vdash \Delta}\ (\wedge L) \qquad \frac{\Gamma \vdash F, F, \Delta}{\Gamma \vdash F, \Delta}\ (contrR) \qquad \frac{\Gamma \vdash \Delta}{\Gamma[\theta] \vdash \Delta[\theta]}\ (Subst)$$

$$\frac{\Gamma \vdash F, \Delta \quad \Gamma, G \vdash \Delta}{\Gamma, F \Rightarrow G \vdash \Delta}\ (\Rightarrow L) \qquad \frac{\Gamma, F, F \vdash \Delta}{\Gamma, F \vdash \Delta}\ (contrL) \qquad \frac{\Gamma, F[\{\overline{x} \mapsto \overline{t}\}] \vdash \Delta}{\Gamma, \forall \overline{x} F \vdash \Delta}\ (\forall L)$$

$$\frac{\Gamma \vdash F, \Delta}{\Gamma \vdash \forall \overline{x} F, \Delta}\ \overline{x} \cap FV(\Gamma \cup \Delta) = \emptyset\ (\forall R) \qquad \frac{\Gamma, F \vdash \Delta}{\Gamma, \exists \overline{x} F \vdash \Delta}\ \overline{x} \cap FV(\Gamma \cup \Delta) = \emptyset\ (\exists L)$$

$$\frac{\Gamma \vdash F, \Delta \quad \Gamma, F \vdash \Delta}{\Gamma \vdash \Delta}\ (Cut) \qquad \frac{\Gamma \vdash F[\{\overline{x} \mapsto \overline{t}\}], \Delta}{\Gamma \vdash \exists \overline{x} F, \Delta}\ (\exists R) \qquad \frac{\Gamma, F \vdash G, \Delta}{\Gamma \vdash F \Rightarrow G, \Delta}\ (\Rightarrow R)$$

Fig. 1: Sequent-based rules for classical first-order logic.

CLKID$^\omega_N$ consists of the rules displayed in Fig. 1, the rules that process equalities from Fig. 2, as well as the 'unfold' and 'case' rules. $(= L)$ is an instance of the corresponding CLKID$^\omega$ rule for which x can also be a non-variable term.

$$\frac{}{\Gamma \vdash t = t, \Delta}\ (= R) \qquad \frac{\Gamma[\{x \mapsto u\}] \vdash \Delta[\{x \mapsto u\}]}{\Gamma, x = u \vdash \Delta}\ x \text{ is not a variable of } u\ (= L)$$

Fig. 2: Sequent-based rules for equality reasoning.

The unfold rule unrolls the definition of the inductive symbol to transform some succedent atom of a sequent. We denote the unfolding of $P(\overline{t}')$ with the production (7), when $P(\overline{t}') \equiv P(\overline{t})[\sigma]$, by

$$\frac{\Gamma \vdash Q_1(\overline{u}_1)[\sigma], \Delta \quad \ldots \quad \Gamma \vdash Q_h(\overline{u}_h)[\sigma], \Delta \qquad \Gamma \vdash P_{i_1}(\overline{t}_1)[\sigma], \Delta \quad \ldots \quad \Gamma \vdash P_{i_l}(\overline{t}_l)[\sigma], \Delta}{\Gamma \vdash P(\overline{t'}), \Delta}\ (R.(7))$$

The *case* rule is a left-introduction operation for inductive predicate symbols:

$$\frac{\text{case distinctions}}{\Gamma, P(s_1, \ldots, s_n) \vdash \Delta}\ (Case\ P)$$

Every production of the form (7) for which $\overline{t} \equiv (t_1, \ldots, t_n)$ produces the *case distinction*

$$\Gamma, s_1 = t_1, \ldots, s_n = t_n, Q_1(\overline{u}_1), \ldots, Q_h(\overline{u}_h), P_{i_1}(\overline{t}_1), \ldots, P_{i_l}(\overline{t}_l) \vdash \Delta \qquad (8)$$

Each variable y from (7) is fresh w.r.t. the *free variables* from the conclusion of the rule (y can be renamed to a fresh variable, otherwise). $P_{i_1}(\overline{t}_1), \ldots, P_{i_l}(\overline{t}_l)$ are *case descendants* of $P(s_1, \ldots, s_n)$.

CLKID$^\omega_N$ pre-proof trees. A *derivation tree* for some sequent S is built by successively applying inference rules starting from S. The terminal nodes in the tree can be either leaves or buds. A *leaf* is labelled by a sequent that is the conclusion of a 0-premise inference rule. A *bud* is every node labelled by a sequent that is the conclusion of no rule. For each bud, there is a *companion*, i.e., an internal node having the same sequent labelling. If a companion is annotated by some sign (e.g., † or $*$), then the buds related to it are uniquely annotated by that sign followed by a number.

Definition 2.1 (pre-proof tree, induction function for tree). *The pair ($\mathcal{D}$, $\mathcal{R}$) denotes a* pre-proof tree *of some sequent S, where $\mathcal{D}$ is a finite derivation tree whose root is labelled by S and $\mathcal{R}$ is a defined* induction function *assigning a companion to every bud in $\mathcal{D}$.*

Example 2.2. A CLKID$^\omega_N$ pre-proof tree of $N(x), N(y) \vdash R(x, y)$ is

$$\dfrac{\dfrac{\dfrac{}{\vdash R(0,0)}\,(R.(9)) \qquad \dfrac{\dfrac{\dfrac{Nx' \vdash R(x',0)\ (\dagger\mathbf{1})}{Nx'' \vdash R(x'',0)}\,(Subst)}{Nx'' \vdash R(sx'',0)}\,(R.(10))}{\underline{\underline{\qquad\qquad}}\,(Case\ N)} \qquad \dfrac{\dfrac{Nx, Ny \vdash R(x,y)\ (*\mathbf{1})}{Nssx', Ny' \vdash R(ssx', y')}\,(Subst)}{\qquad\qquad}\,(Cut)}{\qquad\qquad}$$

$$\dfrac{\dfrac{\dfrac{\underline{Nx'} \vdash R(x',0)\ (\dagger)}{Nx' \vdash R(sx',0)}\,(R.(10)) \qquad \dfrac{Nx', Ny' \vdash R(ssx', y')}{Nx', Ny' \vdash R(sx', sy')}\,(R.(11))}{\underline{\underline{\qquad\qquad}}\,(Case\ N)}}{\qquad\qquad}$$

$$\dfrac{\dfrac{}{Ny \vdash R(0,y)}\,(R.(9)) \qquad \dfrac{\qquad\qquad}{Nx', \underline{Ny} \vdash R(sx', y)}}{\underline{\underline{\qquad\qquad}}\,(Case\ N)}$$

$$\dfrac{\qquad\qquad}{\underline{Nx}, Ny \vdash R(x,y)\ (*)}$$

where the inductive predicate R is defined, as in [8], by the productions

$$\Rightarrow R(0, y) \quad (9) \qquad R(x, 0) \Rightarrow R(sx, 0)\ \ (10) \qquad R(ssx, y) \Rightarrow R(sx, sy)\ \ (11)$$

For lack of horizontal space, we have unambiguously omitted the parentheses and commas when denoting some natural and atom $N(t)$, i.e., $s(t)$ (resp., $N(t)$)

becomes $s\underline{t}$ (resp., $N\underline{t}$), where $\underline{t}$ is the notation of t without parentheses. This alternative notation will be used in the following, when necessary.

The double line means that $(= L)$ was applied on each premise of $(Case)$. The (Cut) premise $Nx' \vdash Nssx'$ is suppressed on the right-hand branch as in Example 6 of [8]. The principal formula for each $(Case)$ application is underlined. Finally, the induction function $\mathcal{R}$ is defined such that the companion of the bud denoted by $(*1)$ (resp., $(\dagger 1)$) is $(*)$ (resp., $(\dagger)$).

Semantics. The semantics for $\mathrm{FOL_{ID}}$ with equality is defined as in [9]. Prefixed points of a monotone operator issued from Φ [1] help to interpret inductive predicates. A standard model for (Σ, Φ) is a first-order structure defined by the least prefixed point, approached by an iteratively built *approximant* sequence.

Definition 2.3 (validity of a sequent). *Let M be a standard model for (Σ, Φ), $\Gamma \vdash \Delta$ a sequent and ρ a valuation which interprets in M the free variables from the sequent. We write $\Gamma \models^M_\rho \Delta$ if whenever G holds in M using ρ, for all $G \in \Gamma$, there is some $D \in \Delta$ that holds in M using ρ. We say that $\Gamma \vdash \Delta$ is M-*true *if $\Gamma \models^M_\rho \Delta$, for every ρ. When M is implicit from the context,* true *is used instead of M-true.*

A rule is *sound*, or preserves the validity, if its conclusion is true whenever its premises are true. Hence, the conclusion of every 0-premise sound rule is true.

Theorem 2.4. *The $CLKID^\omega_N$ inference rules are sound.*

Definition 2.5 (sound pre-proof tree). *A pre-proof tree of a sequent S is* sound *if S is true.*

3 Checking the soundness of pre-proofs

Not every pre-proof tree is sound. A very simple example of unsound pre-proof tree can be built for every false sequent S by firstly adding a copy of some antecedent formula using $(contrL)$ then deleting it using (Wk). Since the resulting sequent is identical to S, its node is a bud. This finishes the pre-proof tree.

We intend to apply an approach similar to that used for building well-founded (Noetherian) induction-based proofs [15] to check the soundness of pre-proof trees. In this setting, a cycle is uniformly represented as a set of paths from root companions to bud nodes. When a node of the pre-proof tree plays the role of bud *and* companion, it is duplicated by some transformation operation such that the roles are played separately by each copy. The normalization process consists in the exhaustive application of the transformation operations that convert a pre-proof tree to a set of pre-proof trees, for short *pre-proof tree-sets*.

In this section, we briefly explain the transformation operations and show how to build digraphs from pre-proof tree-sets. Then, we prove that a pre-proof tree is sound if the minimal cycles from the digraph of the pre-proof tree-set resulting from its normalization satisfy certain ordering and derivability constraints. Finally, we present some strategies for directly building sound pre-proof tree-sets.

3.1 Defining the checking criteria

The first transformation operation applies on an internal node labeled by some premise of $(Subst)$:

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash \Delta}}{\dfrac{\Gamma[\sigma] \vdash \Delta[\sigma]}{\vdots}\,(Subst)} \quad \text{produces} \quad \dfrac{\dfrac{\Gamma \vdash \Delta\ (*1)}{\Gamma[\sigma] \vdash \Delta[\sigma]}\,(Subst)}{\vdots} \qquad \dfrac{\vdots}{\Gamma \vdash \Delta\ (*)}\ \text{(new tree)}$$

The node is duplicated; one of its copies is detached together with the subtree derivation rooted by the node to become a new tree derivation. The two occurrences of the duplicated bud establish a new relation bud-companion by extending the definition of the induction function for trees to sets of trees.

The second transformation rule is performed on a non-root companion of $n-1$ $(n>1)$ buds and annotated by $(*)$, of the form

$$\dfrac{\vdots}{\Gamma \vdash \Delta\ (*)}\ \vdots \quad \text{to give} \quad \Gamma \vdash \Delta\ (*n)\ \vdots \qquad \dfrac{\vdots}{\Gamma \vdash \Delta\ (*)}\ \text{(new tree)}$$

The companion $(*)$ is duplicated such that the subtree derivation rooted by it becomes a new pre-proof tree and the copy of $(*)$ becomes a bud annotated by $(*n)$. A new relation bud-companion is created between $(*n)$ and $(*)$.

Example 3.1. The application of the second transformation on the companion annotated by $(\dagger)$ in the pre-proof tree given in Example 2.2 generates a normalized pre-proof tree-set, as shown in Fig. 3.

$$\dfrac{\dfrac{}{Ny \vdash R(0,y)}\,(R.(9)) \qquad \dfrac{\dfrac{\underline{Nx'} \vdash R(x',0)\ (\dagger\mathbf{1})}{Nx' \vdash R(sx',0)}\,(R.(10)) \quad \dfrac{\dfrac{Nx, Ny \vdash R(x,y)\ (*)}{Nssx', Ny' \vdash R(ssx',y')}\,(Subst)}{Nx', Ny' \vdash R(ssx',y')}\,(Cut)}{Nx', \underline{Ny} \vdash R(sx',y)}\,(R.(11))}{\underline{Nx}, Ny \vdash R(x,y)\ (*)}\,(Case\ N)$$

$$\dfrac{\dfrac{}{\vdash R(0,0)}\,(R.(9)) \qquad \dfrac{\dfrac{Nx' \vdash R(x',0)\ (\dagger)}{Nx'' \vdash R(x'',0)}\,(Subst)}{Nx'' \vdash R(sx'',0)}\,(R.(10))}{\underline{Nx'} \vdash R(x',0)\ (\dagger)}\,(Case\ N)$$

Fig. 3: The normalized pre-proof tree-set.

Definition 3.2 (pre-proof tree-set, induction function for tree-set). *The pair ($\mathcal{MD}$, $\mathcal{MR}$) denotes a* pre-proof tree-set*, where $\mathcal{MD}$ is a multiset of pre-proof trees and $\mathcal{MR}$ is a defined* induction function *assigning a companion to every bud from $\mathcal{MD}$.*

Lemma 3.3. *The normalization process terminates.*

Lemma 3.4. *Let ($\mathcal{MD}$, $\mathcal{MR}$) be the pre-proof tree-set obtained by the normalization of some pre-proof tree of a sequent S. Then, MD*

i) has a pre-proof tree rooted by a node labelled by S, and
ii) is built from pre-proof trees for which the premises of all ($Subst$) rules are bud sequents.

Any pre-proof tree-set can also be represented as a *digraph* of sequents using nodes from its tree-set and arrows annotated with substitutions. Let $S(N)$ denote the sequent labelling N, for every node N. A solid arrow leads a node N^1 to a node N^2 if there is a rule applied on $S(N^1)$ and $S(N^2)$ is a premise of the rule. It is annotated with the *identity substitution* for $S(N)(\equiv \Gamma \vdash \Delta)$, denoted by $\sigma_{id}^{S(N)}$ and defined as $\bigcup_{x \in FV(\Gamma \cup \Delta)} \{x \mapsto x\}$, if N is not a $(= L)$-node. When not ambiguous, the identity substitutions are omitted. If N is a $(= L)$-node whose principal formula is $x = u$, the arrow leaving N is annotated by the *equality substitution* $\{x \mapsto u\}$. On the other hand, a dashed arrow leads a bud B to its companion and is annotated with a substitution written in boldface. If $S(B)$ is the premise of a ($Subst$) rule using the substitution θ, this substitution is θ. Otherwise, it is $\sigma_{id}^{S(B)}$.

For convenience, the sequent $\Gamma \vdash \Delta$ labelling a node N^i in the digraph is indexed by i as $\Gamma \vdash_i \Delta$.

Example 3.5. Fig. 4 shows the digraph of the pre-proof tree-set from Fig. 3.

A *path* is a (potentially infinite) list of nodes built by following the arrows in the digraph. It is ($Subst$)*-free* if none of the sequents labelling its nodes is the premise of some ($Subst$)-rule.

Definition 3.6 (cumulative substitution). *A ($Subst$)-free path $[N^1, \ldots, N^n]$ $(n > 0)$ is annotated by the* cumulative substitution $\sigma_{id}^{all}\sigma_1 \cdots \sigma_{n-1}$*, where σ_i is the substitution annotating the solid arrow leading N_i to N_{i+1}, for each $i \in [1..n-1]$, and σ_{id}^{all} is the* overall *identity substitution $\cup_{N \in [N^1, \ldots, N^n]} \{x \mapsto x \mid x \in FV(\Gamma \cup \Delta)$ and $S(N) \equiv \Gamma \vdash \Delta\}$.*

Example 3.7. The cumulative substitution for the ($Subst$)-free path $[N^1, N^3, N^5, N^6, N^7]$ from Fig. 4 is $\{x \mapsto sx'; y \mapsto sy'\}$, which is the composition of the overall identity substitution $\{x \mapsto x; y \mapsto y\}$ with the substitutions $\{x \mapsto sx'\}$, $\{y \mapsto sy'\}$, and other identity substitutions.

The digraph $\mathcal{P}$ of a pre-proof tree-set ($\mathcal{MD}$, $\mathcal{MR}$) can be partitioned in a set of *strongly connected components* (SCCs). Some of them may have only one

node. The SCCs with more than one node have at least one *cycle*, i.e., a path built along its nodes where the nodes are repeated. A *minimal* cycle does not contain other cycles.

Definition 3.8 (n-cycle). *Every minimal cycle in $\mathcal{P}$ can be represented as a* n-cycle*, defined as a finite circular list $[N_1^1, \ldots, N_1^{p_1}], \ldots, [N_n^1, \ldots, N_n^{p_n}]$ of n (> 0) paths leading root nodes to buds such that $N_{next(i)}^1 = \mathcal{MR}(N_i^{p_i})$, for any $i \in [1..n]$, where $next(i) = 1 + (i \bmod n)$.*

The standard method for checking the soundness of pre-proof trees, e.g. [9], is based on a 'proof by contradiction' approach. Let us assume that the root sequent of some pre-proof tree is false. It is sufficient to show that some global trace condition is satisfied for every infinite path in the pre-proof tree, built by visiting nodes labelled by false sequents if the root sequent is false. This condition stipulates that all successive steps starting from some point in the path are decreasing and certain steps occurring infinitely often are strictly decreasing w.r.t. some semantic ordering underlying ordinals. The trace is a list of *inductive antecedent atoms* (IAAs) of the sequents labelling the nodes from the path. Let $P(\bar{t})$ be one of these atoms. Since P can be generated by a sequence of approximants $(P^\gamma)_{\gamma \geq 0}$, the measure value for $P(\bar{t})$ used by this ordering is the smallest ordinal γ such that $P^\gamma(\bar{t})$ holds for some suitable interpretation. The well-foundedness property of the ordering contradicts the fact that the path is infinite.

The question of whether a pre-proof tree is sound is decidable (see e.g. Proposition 7.4 from [9]), by using a decision procedure based on the automata-based complementation method. On the other hand, the computational and combinatorial complexity of the validation of the global trace condition can be reduced for pre-proof trees of certain structure, e.g., for those having *trace manifolds* [6,7].

The trace manifold condition can be checked only on pre-proof trees in *cycle normal form*, for which the companion of every bud B is also an ancestor of B. Any pre-proof tree can be transformed in a cycle normal form. Mainly, it is unfolded into an infinite pre-proof tree, then the infinite branches are folded to get an equivalent normalized pre-proof tree. An improved complexity bound can be achieved by an iterative 'untangling' process of the pre-proof tree. By Theorem 6.3.6 from [7], if a derivation tree has n nodes, the equivalent normalized derivation tree has no more than $n^{2^{n/2}}$ nodes.

In the same quest to reduce the complexity of the validation process, we adapt the approach for building well-founded induction proofs [15] to generate pre-proof trees that *implicitly* guarantee the validity of the global trace condition. For this, we denote by $\leq_\pi$ a partial ordering defined over the set of instances of every sequent labelling root nodes from some SCC π with cycles. Its strict part is denoted by $<_\pi$ and its equivalence part by $\sim_\pi$. Contrary to [15], $<_\pi$ is not required to be well-founded. We assume that $<_\pi$ is built as the multiset extension of a 'stable under substitutions' ordering $<_a$ defined over IAAs and used to compare instances of the root sequents from π. Hence, $<_\pi$ is also stable under substitutions [2]. We also assume that $\sim_\pi$ is stable under substitutions. Like $<_\pi$,

$<_a$ does not need to be well-founded. For each node N in π and every instance S of $S(N)$, we denote by A_S the measure value (weight) of S, represented as a multiset of IAAs of S and used in the comparisons of S with other sequent instances w.r.t. $\leq_\pi$.

Definition 5.4 of [9] for a trace in a pre-proof tree can be adapted for a digraph $\mathcal{P}$ and simplified to take into account the restricted version of $(= L)$.

Definition 3.9 (trace). *A* trace *following some (potentially infinite) path p in $\mathcal{P}$, denoted by $[N^1, N^2, \ldots]$, is a sequence $(\tau_i)_{(i\geq 0)}$ such that:*

- *$\tau_i = P(\bar{t})$, where $P(\bar{t})$ is an IAA of $S(N^i)$;*
- *if $\Gamma_i \vdash \Delta_i$ is the conclusion of $(Subst)$ then $\tau_i = \tau_{i+1}[\rho]$, where ρ is the substitution associated with the rule instance;*
- *if $\Gamma_i, x = u \vdash \Delta_i$ is the conclusion of $(= L)$, then $\tau_{i+1} = \tau_i[\{x \mapsto u\}]$;*
- *if $S(N^i)$ is the conclusion of a $(Case)$-rule, then either i) $\tau_{i+1} = \tau_i$, or ii) τ_i is its principal formula and τ_{i+1} is a case descendant of τ_i. In this case, i is called a* progression point*;*
- *if $S(N^i)$ is the conclusion of any other rule, then $\tau_{i+1} = \tau_i$.*

We say that an IAA τ_j *derives from* an IAA τ_i using the trace $(\tau_k)_{(k\geq 0)}$ if $i < j$. Given two arbitrary substitutions γ and δ, we also say that $\tau_j[\gamma]$ derives from $\tau_i[\delta]$ using $(\tau_k)_{(k\geq 0)}$. Let π be a SCC from $\mathcal{P}$ and $<_a$ the 'stable under substitutions' ordering defined over the set of instances of the IAAs from the root sequents inside π. We can define $<_\pi$ as the multiset extension of $<_a$ that satisfies some derivability constraints.

Definition 3.10 ($<_\pi$-derivability). *Let N^i and N^j be two nodes occurring in some path p from an SCC π, θ and δ two substitutions, and $A'_{S(N^i)[\theta]}$ (resp., $A'_{S(N^j)[\delta]}$) the multiset resulting from $A_{S(N^i)[\theta]}$ (resp., $A_{S(N^j)[\delta]}$) after the pairwise deletion of all common IAAs from $A_{S(N^i)[\theta]}$ and $A_{S(N^j)[\delta]}$. Then, $S(N^j)[\delta]$ is* $<_\pi$-derivable *from $S(N^i)[\theta]$ along p if i) for each $l \in A'_{S(N^j)[\delta]}$, there exists $l' \in A'_{S(N^i)[\theta]}$ such that $l' >_a l$ and l derives from l' using some trace following p, and ii) for each $l \in A_{S(N^j)[\delta]} \backslash A'_{S(N^j)[\delta]}$, there is some $l' \in A_{S(N^i)[\theta]} \backslash A'_{S(N^i)[\theta]}$ such that $l \equiv l'$ and l derives from l' using some trace following p.*

Lemma 3.11. *In Definition 3.10, for each IAA l from $S(N^j)[\delta]$ there is an IAA l' from $S(N^i)[\theta]$ such that l derives from l' using some trace following p.*

We give below some useful properties of the $<_\pi$-derivability relation.

Lemma 3.12. *$S <_\pi S'$ if S is $<_\pi$-derivable from S' along some path p in π.*

Lemma 3.13. *The '$<_\pi$-derivability' relation is stable under substitutions. It is also transitive, i.e., if S is $<_\pi$-derivable from S' along p and S' is $<_\pi$-derivable from S'' along p then S is $<_\pi$-derivable from S'' along p, for some path p in π.*

We are ready to introduce the ordering constraints that help to discharge induction hypotheses by n-cycles.

Definition 3.14 (induction hypothesis (IH), IH discharged by an n-cycle, IH-node). *Let π be an SCC with cycles and C an n-*cycle *$[N_1^1, \ldots, N_1^{p_1}], \ldots, [N_n^1, \ldots, N_n^{p_n}]$ from π. The instances $S(N_j^{p_j})[\delta_j]$ ($j \in [1..n]$) are called* induction hypotheses *(IHs), where δ_j annotates the dashed arrow starting from $N_j^{p_j}$ in C. For all $i \in [1..n]$, let θ_i^c be the cumulative substitution annotating $[N_i^1, \ldots, N_i^f]$, where the* IH-node *N_i^f is either i) $N_i^{p_i}$ if $[N_i^1, \ldots, N_i^{p_i}]$ is ($Subst$)-free, or ii) $N_i^{p_i-1}$, otherwise. The IHs $S(N_j^1)[\delta_j]$ ($j \in [1..n]$) are* discharged *by C if, $\forall i \in [1..n]$, $S(N_i^{p_i})[\delta_i]$ is $<_\pi$-derivable from $S(N_i^1)[\theta_i^c]$ along $[N_i^1, \ldots, N_i^{p_i}]$.*

Definition 3.14 is well-formed; the cumulative substitution can be computed for the case ii) because, for each $i \in [1..n]$, $[N_i^1, \ldots, N_i^{p_i-1}]$ is a ($Subst$)-free path, by following the claim ii) of Lemma 3.4. By construction, $S(N_i^f)$ is the IH $S(N_{next(i)}^1)[\delta_{next(i)}]$, for all $i \in [1..n]$. For every IAA l of a sequent bud corresponding to some τ_i from a trace $(\tau_k)_{(k\geq 0)}$ following a path from some minimal cycle, we define the *history* of l as the subtrace $(\tau_k)_{(k<i)}$. If each such IAA stores its history during the proof construction, every derivability constraint can be decided in linear time w.r.t. the size of the history, by visiting one by one each element in the history.

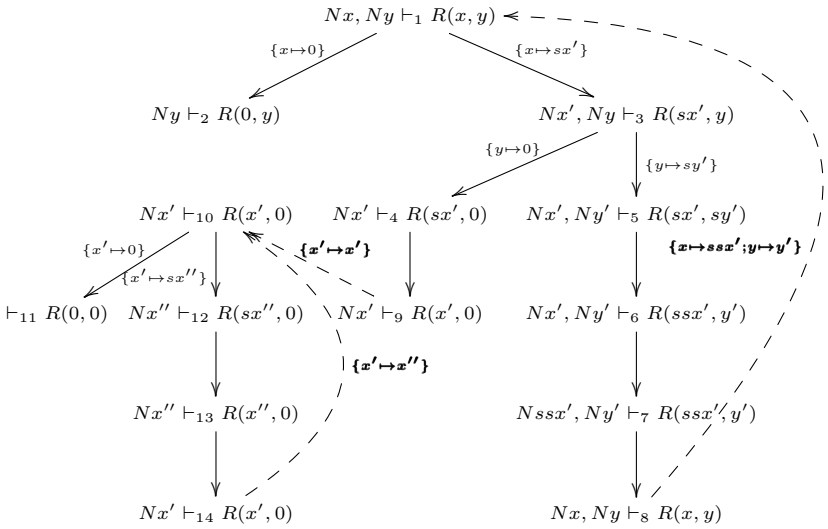

Fig. 4: The digraph of the pre-proof tree-set from Fig. 3.

Example 3.15. The digraph from Fig. 4 has two SCCs with cycles. One of them, denoted by π, consists of the 1-cycle $[N^{10}, N^{12}, N^{13}, N^{14}]$. The other, denoted by π', is made of the 1-cycle $[N^1, N^3, N^5, N^6, N^7, N^8]$. No cycle has the bud N^9.

We define the measure value for any sequent instance of the form $S(N^{10})[\{x' \mapsto t; \ldots\}]$ (resp., $S(N^1)[\{x \mapsto t_1; y \mapsto t_2; \ldots\}]$) as $\{N(t)\}$ (resp., $\{N(t_2)\}$). $<_\pi$ and $<_{\pi'}$ are multiset extensions of an ordering $<_a$ over the IAAs

of the form $N(t)$ and whose measure value is $\{t\}$, for any term t. $<_a$ is the multiset extension of the 'stable under substitutions' rpo ordering [2], denoted by $<_{rpo}$ and based on the symbol precedence establishing that 0 is smaller than s.

Every IH from these 1-cycles is discharged. The IH $S(N^{10})[\{x' \mapsto x''\}]$ is $<_\pi$-derivable from $S(N^{10})[\{x' \mapsto sx''\}]$ along the $(Subst)$-free path $[N^{10}, N^{12}, N^{13}, N^{14}]$, by using the trace $[Nx', Nx'', Nx'', Nx']$ for the IAA Nx' of $S(N^{10})$. Similarly, the IH $S(N^1)[\{x \mapsto ssx'; y \mapsto y'\}]$ is $<_{\pi'}$-derivable from $S(N^1)[\{x \mapsto sx'; y \mapsto sy'\}]$ along the $(Subst)$-free path $[N^1, N^3, N^5, N^6, N^7, N^8]$, by using the trace $[Ny, Ny, Ny', Ny', Ny', Ny]$ for the IAA Ny of $S(N^1)$. The two comparisons hold as $z <_{rpo} sz$, for every variable z.

Definition 3.16 (proof). *A* proof *is every pre-proof tree-set whose digraph has only n-cycles that discharge their IHs.*

Theorem 3.17 (soundness). *The root sequents from every proof are true.*

Proof (Sketch). It follows the general structure of the soundness proof for some cyclic well-founded inference systems, e.g., see Theorem 5.11 from [17]. Mainly, we define a partial ordering $<_\mathcal{R}$ over the root nodes from the digraph $\mathcal{P}$ of a proof such that, for every two distinct root nodes N^1 and N^2, we have $N^1 <_\mathcal{R} N^2$ if i) N^1 and N^2 are not in the same SCC, and ii) N^1 can be joined from N^2 in $\mathcal{P}$.

By contradiction, we assume that there exists a root node N such that $S(N)$ is false. A classical induction reasoning using $<_\mathcal{R}$ allows to explore all possibilities for N to be considered as one of the root nodes from $\mathcal{P}$. The most difficult case is when N is part of some SCC with cycles. A contradiction yields by showing that there exists a trace with an infinite number of progression points, using similar arguments as in the proof of Lemma 5.7 [9] and witnessed by an infinite strictly decreasing sequence of ordinals, thanks to the $<$-derivability constraints. □

Theorem 3.17 is the key argument for proving that our approach allows indeed to verify pre-proof trees.

Lemma 3.18. *A pre-proof tree is sound if the pre-proof tree-set resulting from its normalization operation is a proof.*

Validation costs. We analyse the worst-case time complexity for validating the soundness of a pre-proof tree of n nodes with p ($< n$) buds occurring in minimal cycles. The number of transformation operations during the normalization step is given by the sum of non-root companions and non-terminal $(Subst)$-nodes, which is smaller than $2n$. The cost of a transformation operation, including the node duplication and the creation of a bud-companion relation, is assumed to be some constant c. Hence the cost of the transformation operations is smaller than $2nc$. If c' is the constant representing the cost for annotating a substitution, the cost for building the digraph of the normalized pre-proof tree-set is smaller than nc'. The partition of a digraph in SCCs can be done in linear time [18].

If B denotes a bud occurring in a minimal cycle, the IH that instantiates $S(B)$ is unique because B has only one companion and at most one companion can be the root of the tree including B. The number of $<$-derivability constraints

is that of their buds, i.e., p. In the worst case, p is $n-1$. The validation cost of a $<$-derivability constraint is the sum of the costs of derivability and ordering constraints. The time complexity for checking whether an IAA l derives from another IAA l' is linear w.r.t. the size of the history of l, which is a value smaller than n. The time complexity for checking a multiset extension relation is $O(rq)$, where r and q are the number of IAAs from the measure value of the compared sequents. In the worst case, when all bud IAAs have their history of length n and p is $n-1$, the time complexity for checking the derivability constraints is $O(n^2k^2)$, where k is the maximal cardinality of a sequent's measure value. Similarly, the worst-case validation cost of the ordering constraints is polynomial in k, the maximal size of a literal and n, if the time complexity for comparing two IAAs is at most polynomial, for example by using a Knuth-Bendix ordering [2].

3.2 Strategies for directly building proofs

Theorem 3.17 suggests that sequents can also be proved by directly building sound pre-proof tree-sets. For this purpose, we adapt the DRaCuLa strategy [15]. Mainly, the trees from a pre-proof tree-set are developed by applying the CLKID$^\omega_N$ rules, as usual, with the following exceptions:

- when applying a ($Subst$)-rule, the premise becomes a bud sequent, as shown for the first transformation of the normalization procedure. The next step is to develop a new tree rooted by the companion of the bud;
- when a bud is about to be created, several scenarios may happen. As a preliminary step, if its companion is a non-root node the second transformation of the normalization procedure is applied. If the bud candidate is part of an n-cycle that discharges its IHs, the bud is created (scenario 1). If it is not yet the case, either i) the strategy tries to build an n-cycle, by developing parts from other trees (scenario 2), or ii) the n-cycle does not discharge its IHs (scenario 3); in this case, a backtracking step is required either to redefine the ordering at the SCC level, or to redo previous steps, or to continue to develop the proof by applying a CLKID$^\omega_N$ rule on the sequent labelling the bud candidate.
  For (scenario 1), not only the current bud candidate is created, but all the bud candidates from the n-cycle are built, hence *simultaneous* induction is performed.

Example 3.19. The above strategy can build the pre-proof tree from Example 2.2. The progression in its construction can be retraced by following the indexes of the sequents from the digraph in Fig. 4.

This proof strategy uses heuristics based on ordering constraints, different from the iterative depth-first search heuristics used by Cyclist.

Example 3.20. One could have built a new bud of (*) from the pre-proof tree of Example 2.2, by developing (†) such that $N0$ is added as IAA, then ($Subst$) applied conveniently. The new 1-cycle from its digraph is part of the SCC π' of

the digraph from Example 3.15. However, it cannot discharge its IH because the induction ordering is now $<_{\pi'}$ instead of $<_{\pi}$.

Sound pre-proof trees can also be directly generated to satisfy implicitly the ordering constraints, similar to implicit induction proofs, by using a *reductive* proof strategy based on a unique induction ordering $<$. Such strategy guarantees that, for every two successive nodes N^i and N^{i+1} from each path p, of the form $[N^1, \ldots, N^n]$ and occurring in the definition of some minimal cycle of its digraph, and $i \in [1..f-1]$, we have either $A_{S(N^{i+1})[\theta^c_{i+1}]} \equiv A_{S(N^i)[\theta^c_i]}$ or $S(N^{i+1})[\theta^c_{i+1}]$ is $<$-derivable from $S(N^i)[\theta^c_i]$ along p, where f is the index of the IH-node in $[N^1, \ldots, N^n]$ and θ^c_j is the cumulative substitution for the path $[N^j, \ldots, N^f]$ ($j \in [1..f]$). The derivability constraints are satisfied if the syntactic equality relation is not satisfied at least once along p. Indeed, knowing that the $<$-derivability relation is transitive (Lemma 3.13), we have that $S(N^f)$ is $<$-derivable from $S(N^1)[\theta^c_1]$ along p, as required. If the rule applied at step i is different from $(= L)$, we have that $\theta^c_i \equiv \theta^c_{i+1}$. In this case, it is sufficient to ensure instead that $A_{S(N^{i+1})} \equiv A_{S(N^i)}$ or $S(N^{i+1})$ is $<$-derivable from $S(N^i)$ along p, due to the 'stability under substitutions' property of $<$-derivability (again Lemma 3.13).

Example 3.21. As a proof of concept, we define the derived rule $(DCase)$:

$$\frac{S_1 \quad \ldots \quad S_n}{\Gamma, P(\overline{x}) \vdash \Delta}\,(DCase\ P) \quad \text{as} \quad \frac{\dfrac{\dfrac{S_1}{\text{case distinction}}\,(=L) \quad \ldots \quad \dfrac{S_n}{\text{case distinction}}\,(=L)}{\Gamma, P(\overline{x}), \underline{P(\overline{x})} \vdash \Delta}\,(Case\ P)}{\Gamma, P(\overline{x}) \vdash \Delta}\,(contrL)$$

where $\overline{x}$ is a vector of variables. We also define the (Bud) rule:

$$\frac{\text{(bud sign)}}{\Gamma \vdash \Delta}\,(Bud) \qquad \text{as} \qquad \frac{\dfrac{\Gamma' \vdash \Delta' \text{ (bud sign)}}{\Gamma'[\sigma] \vdash \Delta'[\sigma]}\,(Subst)}{\Gamma \vdash \Delta}\,(Wk)$$

if $\Gamma' \vdash \Delta'$ *subsumes* $\Gamma \vdash \Delta$ with substitution σ, i.e., $\Gamma'[\sigma] \subseteq \Gamma$ and $\Delta'[\sigma] \subseteq \Delta$. Different variants of the subsumption operation are widely employed by the current theorem provers, CYCLIST being one of them.

By using the alternative notation without parentheses, a pre-proof of $Nx, Ny \vdash Q(x, y)$ is built below by firstly trying to apply the unfold rules followed by (Bud), then (Del) and, finally, $(DCase)$. (Del) is a restricted version of the (Wk) rule that deletes the IAAs of the form $N(t)$ if none of the inductive succedent atoms from the conclusion has t as argument. It can be noticed that the history of every IAA occurring in each premise of any rule r from the above rules but (Bud) has one of the IAAs from the conclusion of r.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{N0 \vdash P0}\,(R.(1)) \quad \dfrac{\dfrac{(*1)}{Nsz, Nz \vdash Pz}\,(Bud) \quad \dfrac{(\dagger 2)}{Nz, Nsz \vdash Qzsz}\,(Bud)}{Nsz, Nz \vdash Psz}\,(R.(2))}{Nx \vdash Px\ (*)}\,(DCase\ N)}{Nx, Nz, Nsz \vdash Px}\,(Del) \quad \dfrac{(\dagger 1)}{Nx, Nz, Nsz \vdash Qxz}\,(Bud)}{Nx, N0 \vdash Qx0}\,(R.(3)) \quad \dfrac{}{Nx, Nz, Nsz \vdash Qxsz}\,(R.(4))}{Nx, \underline{Ny} \vdash Qxy\ (\dagger)}\,(DCase\ N)$$

The proof strategy is reductive if the measure value for each sequent of the form $\Gamma, N(t) \vdash P(t)$ (resp., $\Gamma, N(t_1), N(t_2) \vdash Q(t_1, t_2)$ is the multiset of IAAs $\{N(t), N(t)\}$ (resp., $\{N(t_1), N(t_1), N(t_2)\}$) and $<$ is defined as the multiset extension of the ordering $<_a$ over IAAs from Example 3.15. It can be checked that the $<$-derivability constraints are satisfied, by taking into account that the unique SCC with cycles of the digraph associated to its normalized pre-proof tree-set is built from the union of two 1-cycles, $[(*), \ldots, (*1)]$ and $[(\dagger), \ldots, (\dagger 1)]$, and one 2-cycle $[(*), \ldots, (\dagger 2)], [(\dagger), \ldots, \text{copy of } (*)]$.

By Theorem 3.17, our approach allows to prove several conjectures *simultaneously*. This is a feature specific to formula-based Noetherian induction reasoning [15] as that employed by the implicit induction inference systems. It is particularly useful when the proofs of the conjectures are mutually dependent.

Example 3.22. The normalization step for the pre-proof tree from Example 3.21 can be avoided if the pre-proof trees of $Nx \vdash Px$ and $Nx, Ny \vdash Qxy$ are developed simultaneously.

4 Conclusions and future work

We have presented a new method to validate a class of CLKID$^\omega$ pre-proof trees by converting them to pre-proof tree-sets, then showing that the global trace condition is implicitly satisfied if some ordering and derivability constraints hold. Every infinite path p from a pre-proof tree normalized to a proof (tree-set) can be built by concatenating path segments from the definition of the minimal cycles of its proof. These constraints ensure that there is an infinitely progressing trace following some tail of p. Our approach allows more flexibility; a different induction ordering can be defined for each SCC with cycles from the digraph of the proof. This is not the case from the unique induction ordering defined over the buds of a pre-proof tree with trace manifolds [6,7]. Also, our approach does not require pre-proof trees to be in cycle normal form that are, in the worst case, exponentially bigger.

The soundness check can be done in polynomial time provided that the time complexity for comparing two IAAs is at most polynomial. We defined proof strategies ensuring that the number of ordering constraints equals that of the induction hypotheses that are really required in the proof. In practice, their number is generally small even for proofs concerning real-size applications. For example, every cyclic induction proof from [15] (see Table 1) includes at most 8 induction hypotheses and 4 minimal cycles.

The ordering constraints are implicitly satisfied by reductive proof strategies. In the future, we plan to define new (derived) rules and proof strategies that *automatically* generate more compact reductive proof derivations and provide a better control of the proof development. The main challenge of our approach remains to find the 'good' induction orderings.

References

1. P. Aczel. An introduction to inductive definitions. In J. Barwise, editor, *Handbook of Mathematical Logic*, pages 739–782. North Holland, 1977.
2. F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.
3. G. Barthe and S. Stratulat. Validation of the JavaCard platform with implicit induction techniques. In R. Nieuwenhuis, editor, *RTA (Rewriting Techniques and Applications)*, volume 2706 of *LNCS*, pages 337–351. Springer, 2003.
4. A. Bouhoula and M. Rusinowitch. Implicit induction in conditional theories. *Journal of Automated Reasoning*, 14(2):189–235, 1995.
5. F. Bronsard, U.S. Reddy, and R. Hasker. Induction using term orderings. In *CADE (Conf. on Automated Deduction)*, volume 814 of *LNCS*, pages 102–117. Springer, 1994.
6. J. Brotherston. Cyclic proofs for first-order logic with inductive definitions. In *Proceedings of TABLEAUX-14*, volume 3702 of *LNAI*, pages 78–92. Springer-Verlag, 2005.
7. J. Brotherston. *Sequent Calculus Proof Systems for Inductive Definitions*. PhD thesis, University of Edinburgh, November 2006.
8. J. Brotherston, N. Gorogiannis, and R. L. Petersen. A generic cyclic theorem prover. In *APLAS-10 (10th Asian Symposium on Programming Languages and Systems)*, volume 7705 of *LNCS*, pages 350–367. Springer, 2012.
9. J. Brotherston and A. Simpson. Sequent calculi for induction and infinite descent. *Journal of Logic and Computation*, 21(6):1177–1216, 2011.
10. G. Gentzen. Untersuchungen über das logische Schließen. I. *Mathematische Zeitschrift*, 39:176–210, 1935.
11. O. Kupferman and M. Vardi. Weak alternating automata are not that weak. *ACM Transactions on Computational Logic (TOCL)*, 2(3):408–429, 2001.
12. M. Michel. Complementation is more difficult with automata on infinite words. Technical report, CNET, 1988.
13. S. Negri and J. v. Plato. *Structural Proof Theory*. Cambridge University Press, 2001.
14. M. Rusinowitch, S. Stratulat, and F. Klay. Mechanical verification of an ideal incremental ABR conformance algorithm. *Journal of Automated Reasoning*, 30(2):53–177, 2003.
15. S. Stratulat. A unified view of induction reasoning for first-order logic. In A. Voronkov, editor, *Turing-100 (The Alan Turing Centenary Conference)*, volume 10 of *EPiC Series*, pages 326–352. EasyChair, 2012.
16. S. Stratulat. Structural vs. cyclic induction: a report on some experiments with Coq. In *SYNASC (International Symposium on Symbolic and Numeric Algorithms for Scientific Computing)International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, pages 27–34. IEEE Computer Society, 2016.
17. S. Stratulat. Mechanically certifying formula-based Noetherian induction reasoning. *Journal of Symbolic Computation*, 80, Part 1:209–249, 2017.
18. R. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146—160, 1972.
19. The Coq development team. *The Coq Reference Manual*. INRIA, 2017.
20. C.-P. Wirth. Descente infinie + Deduction. *Logic Journal of the IGPL*, 12(1):1–96, 2004.