

Coinductive Techniques for Checking Satisfiability of Generalized Nested Conditions

Lara Stoltenow, Barbara König ✉

University of Duisburg-Essen, Germany

Sven Schneider ✉

Hasso Plattner Institute at the University of Potsdam, Germany

Andrea Corradini ✉

Università di Pisa, Italy

Leen Lambers ✉

Brandenburgische Technische Universität Cottbus-Senftenberg, Germany

Fernando Orejas ✉

Universitat Politècnica de Catalunya, Spain

Abstract

We study nested conditions, a generalization of first-order logic to a categorical setting, and provide a tableau-based (semi-decision) procedure for checking (un)satisfiability and finite model generation. This generalizes earlier results on graph conditions. Furthermore we introduce a notion of witnesses, allowing the detection of infinite models in some cases. Paths in a tableau must be fair, where fairness requires that all relevant parts of a condition are processed eventually. Since the correctness arguments are non-trivial, we rely on coinductive proof methods and up-to techniques that simplify and structure the arguments. We distinguish between two types of categories: categories where all sections are isos, allowing for a simpler tableau calculus that includes finite model generation. In the general case where this requirement is lifted, model generation does not work anymore, but we still obtain a sound and complete calculus.

1 Introduction

Nested graph conditions are a well-known specification technique for graph transformation systems [7] and have been shown to be equivalent to first-order predicate logic (FOL) in [22, 7]. Their definition however is quite different from FOL and they are naturally equipped with operations such as shift, a form of partial evaluation, which is difficult to specify directly in FOL. Furthermore they are well-suited for specifying application conditions in graph rewriting and the computation of weakest preconditions and strongest postconditions wrt. graph rewriting rules can be described in a natural way [1], using shift.

In [1] it has also been observed that graph conditions can be generalized to the categorical setting of reactive systems [15] and that FOL is only one instance of graph conditions, by using graphs and injective graph morphisms as a category. Other specializations can be obtained by instantiating to other categories, one prominent example being cospan categories where the graphs, equipped with an inner and an outer interface, are the arrows. This opens up the treatment of generic specification languages.

Here we are in particular interested in satisfiability checks on the general categorical level. As in FOL, satisfiability can be an undecidable problem (depending on the category), and we propose a semi-decision procedure that can simultaneously serve as a model finder. For FOL there are several well-known methods for satisfiability checking, using for instance resolution or tableau proofs [5], while model generation is typically done separately. The realization that satisfiability checking is feasible directly on graph conditions came in [17, 18], and a set of tableau rules [17] was presented without a proof of (refutational) completeness that was



© L. Stoltenow, B. König, S. Schneider, A. Corradini, L. Lambers and F. Orejas;
licensed under Creative Commons License CC-BY 4.0

Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

later published in [12], together with a model generation procedure [25]. A generalization to non-injective graph morphisms was presented in [16].

The contributions of the current paper can be summarized as follows:

▷ We generalize the tableau-based semi-decision procedure for graph conditions from [12] to the level of general categories, under some mild constraints (such as the existence of so-called representative squares [1]). We present a procedure that has some resemblance to the construction of a tableau in FOL.

▷ We distinguish between two cases: one simpler case in which all sections (arrows that have a right inverse) in the category under consideration are isomorphisms (Section 3); and a more involved case where this does not necessarily hold (Section 5). For the former case we can give additional guarantees, such as model generation whenever there exists a so-called finitely decomposable model, generalizing the notion of finite models. The results generalize [12, 16, 25] from graphs and graph morphisms to an abstract categorical level.

▷ The completeness argument for the satisfiability checking procedure – in particular showing that non-termination implies the existence of an infinite model – requires that the tableau construction satisfies a fairness constraint. The resulting proof is non-trivial and – compared to the proof in [12] – we show that it can be reformulated using up-to techniques. Here we give it a completely new and hopefully clarifying structure that relies on coinductive methods [19, 21].

▷ Furthermore we can use coinductive techniques to display witnesses for infinite models (Section 4): in some cases where only infinite models exist and hence the tableau construction is non-terminating, we are still able to stop and determine that there does exist a model.

Coinductive techniques [23] are reasoning techniques based on greatest fixpoints, which are in particular suitable to analyze infinite or cyclic structures. Such techniques can be enhanced with so-called up-to techniques [21]. To the best of our knowledge, such techniques have not yet been employed in the context of FOL satisfiability checking.

2 Preliminaries

2.1 Coinductive Techniques

A *complete lattice* is a partially ordered set (L, \sqsubseteq) where each subset $Y \subseteq L$ has a greatest lower bound, denoted by $\bigsqcap Y$ and a least upper bound, denoted by $\bigsqcup Y$.

A function $f: L \rightarrow L$ is *monotone* if for all $l_1, l_2 \in L$, $l_1 \sqsubseteq l_2$ implies $f(l_1) \sqsubseteq f(l_2)$, *idempotent* if $f \circ f = f$, and *extensive* if $l \sqsubseteq f(l)$ for all $l \in L$.

Given a monotone function $f: L \rightarrow L$ we are in particular interested in its *greatest fixpoint* νf . By Tarski's Theorem [27], $\nu f = \bigsqcup \{x \mid x \sqsubseteq f(x)\}$, i.e., the greatest fixpoint is the least upper bound of all post-fixpoints. Hence for showing that $l \sqsubseteq \nu f$ (for some $l \in L$), it is sufficient to prove that l is below some post-fixpoint l' , i.e., $l \sqsubseteq l' \sqsubseteq f(l')$.

The idea of using up-to techniques is to define a monotone function $u: L \rightarrow L$ (the up-to function) and check whether u is *f-compatible*, that is $u \circ f \sqsubseteq f \circ u$. If that holds every post-fixpoint l of $f \circ u$ (that is $l \sqsubseteq f(u(l))$) is below the greatest fixpoint of f ($l \sqsubseteq \nu f$). This simple technique can often greatly simplify checking whether a given element is below the greatest fixpoint. In addition it always holds that $u(\nu f) = \nu f$. For more details see [19].

2.2 Categories

We will use standard concepts from category theory. Given an arrow $f: A \rightarrow B$, we write $\text{dom}(f) = A$, $\text{cod}(f) = B$. For two arrows $f: A \rightarrow B$, $g: B \rightarrow C$ we denote their composition

by $f;g: A \rightarrow C$. An arrow $s: A \rightarrow B$ is a *section* (also known as *split mono*) if there exists $r: B \rightarrow A$ such that $s;r = \text{id}$. That is, sections are those arrows that have a right-inverse. Arrows that have a left-inverse (in this case r) are called *retractions*.

As in graph rewriting we will consider the category $\mathbf{Graph}_{\text{fin}}$, which has finite graphs as objects and graph morphisms as arrows. We also consider $\mathbf{Graph}_{\text{fin}}^{\text{inj}}$, the subcategory of $\mathbf{Graph}_{\text{fin}}$ that has the same objects, but only injective, i.e. mono, graph morphisms. In this category the sections are exactly the isos, while this is not the case in $\mathbf{Graph}_{\text{fin}}$.

Another important example category that will be used in Section 4 is based on cospans: note that reactive systems instantiated with cospans [10, 24, 26] yield exactly double-pushout rewriting [4]. Given a base category \mathbf{D} with pushouts, the category $\mathbf{Cospan}(\mathbf{D})$ has as objects the objects of \mathbf{D} and as arrows *cospans*, which are equivalence classes of pairs of arrows of the form $A \xrightarrow{f_L} X \xleftarrow{f_R} B$, where the middle object is considered up to isomorphism. Cospan composition is performed via pushouts (for details see Appendix A).

A cospan is *left-linear* if its left leg f_L is mono. For adhesive categories [11], the composition of left-linear cospans again yields a left-linear cospan, and $\mathbf{ILC}(\mathbf{D})$ is the subcategory of $\mathbf{Cospan}(\mathbf{D})$ where the arrows are restricted to left-linear cospans.

Note that $\mathbf{Graph}_{\text{fin}}^{\text{inj}}$ can be embedded into $\mathbf{ILC}(\mathbf{Graph}_{\text{fin}})$ by transforming a graph morphism f to a cospan with f as the left leg and the identity as the right leg.

2.3 Generalized Nested Conditions

We consider (nested) conditions over an arbitrary category \mathbf{C} in the spirit of reactive systems [15, 14]. Following [22, 7], we define conditions as finite tree-like structures, where nodes are annotated with quantifiers and objects, and edges are annotated with arrows.

► **Definition 1** (Condition). *Let \mathbf{C} be a category. The set of conditions $\text{Cond}(A)$ over an object A is defined inductively as the set of all conditions \mathcal{A} , where \mathcal{A} is either*

- *a finite conjunction of universals $\bigwedge_{i \in \{1, \dots, n\}} \forall f_i. \mathcal{A}_i = \forall f_1. \mathcal{A}_1 \wedge \dots \wedge \forall f_n. \mathcal{A}_n$, or*
- *a finite disjunction of existentials $\bigvee_{i \in \{1, \dots, n\}} \exists f_i. \mathcal{A}_i = \exists f_1. \mathcal{A}_1 \vee \dots \vee \exists f_n. \mathcal{A}_n$*

where $f_i: A \rightarrow A_i$ are arrows and $\mathcal{A}_i \in \text{Cond}(A_i)$ are conditions. We call $A = \text{RO}(\mathcal{A})$ the root object of the condition \mathcal{A} . Each subcondition $\mathcal{Q}f_i. \mathcal{A}_i$ ($\mathcal{Q} \in \{\forall, \exists\}$) is called a child of \mathcal{A} . The constants true (empty conjunction) and false (empty disjunction) serve as the base cases.

The definition of conditions requires that conjunctions contain only universal children and disjunctions only existential children. However, conjunctions of existential conditions \mathcal{A}, \mathcal{B} can be expressed as $\forall \text{id}. \mathcal{A} \wedge \forall \text{id}. \mathcal{B}$, and similarly for disjunctions of universals. Hence we sometimes write $\mathcal{A} \wedge \mathcal{B}$ or $\mathcal{A} \vee \mathcal{B}$ for arbitrary conditions in the proofs.

Instantiated with $\mathbf{Graph}_{\text{fin}}$ respectively $\mathbf{Graph}_{\text{fin}}^{\text{inj}}$, conditions are equivalent to graph conditions as defined in [7], and equivalence to first-order logic has been shown in [22].

While in [1] a model for a condition was a single arrow, we have to be more general, since there are some satisfiable conditions that have no finite models. In particular, we evaluate conditions on infinite sequences of arrows $\bar{a} = [a_1, a_2, a_3, \dots]$, where $A \xrightarrow{a_1} A_1 \xrightarrow{a_2} A_2 \xrightarrow{a_3} \dots$, called *composable sequences*.¹ We define $\text{dom}(\bar{a}) = \text{dom}(a_1)$ and we call such a sequence *finite* iff for some index k all a_i with $i > k$ are identities.

► **Definition 2** (Satisfaction). *Let $\mathcal{A} \in \text{Cond}(A)$. Let $\bar{a} = [a_1, a_2, a_3, \dots]$ be a composable sequence with $A = \text{dom}(\bar{a})$. We define the satisfaction relation $\bar{a} \models \mathcal{A}$ as follows:*

¹ Another option would be to work in the category of potentially infinite graphs. However, that would allow conditions based on infinite graphs for which satisfiability checks become algorithmically infeasible.

131 ■ $\bar{a} \models \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i$ iff for every $i \in I$ and every arrow $g: \text{RO}(\mathcal{A}_i) \rightarrow B$ and all $n \in \mathbb{N}_0$ we
 132 have: if $a_1; \dots; a_n = f_i; g$, then $[g, a_{n+1}, \dots] \models \mathcal{A}_i$.

133 ■ $\bar{a} \models \bigvee_i \exists f_i. \mathcal{A}_i$ iff there exists $i \in I$ and an arrow $g: \text{RO}(\mathcal{A}_i) \rightarrow B$ and some $n \in \mathbb{N}_0$ such
 134 that $a_1; \dots; a_n = f_i; g$ and $[g, a_{n+1}, \dots] \models \mathcal{A}_i$.

135 Note that this correctly covers the base cases ($\bar{a} \models \text{true}$, $\bar{a} \not\models \text{false}$ for every sequence \bar{a}).
 136 Furthermore $a_1; \dots; a_n$ equals the identity whenever $n = 0$.

137 For a finite sequence $\bar{a} = [a_1, \dots, a_k, \text{id}, \text{id}, \dots]$ this means that $\bar{a} \models \mathcal{A}$ iff $a = a_1; \dots; a_k$ is
 138 a model for \mathcal{A} according to the definition of satisfaction given in [1]. In this case we write
 139 $[a_1, \dots, a_k] \models \mathcal{A}$ or simply $a \models \mathcal{A}$.

140 ► **Remark 3.** In the following we use **Cond** to denote the set² of all conditions and **Seq** as the
 141 set of all composable sequences of arrows (potential models). ┘

142 We write $\mathcal{A} \models \mathcal{B}$ (\mathcal{A} implies \mathcal{B}) if $\text{RO}(\mathcal{A}) = \text{RO}(\mathcal{B})$ and for every \bar{a} with $\text{dom}(\bar{a}) = \text{RO}(\mathcal{A})$
 143 we have: if $\bar{a} \models \mathcal{A}$, then $\bar{a} \models \mathcal{B}$. Two conditions are equivalent ($\mathcal{A} \equiv \mathcal{B}$) if $\mathcal{A} \models \mathcal{B}$ and $\mathcal{B} \models \mathcal{A}$.

144 Every condition can be transformed to an equivalent condition that alternates between
 145 \forall, \exists by inserting $\forall \text{id}$ or $\exists \text{id}$ as needed. Such conditions are called *alternating*.

146 We also define what it means for two conditions to be isomorphic by requiring that the
 147 two trees are connected by isos on every level, forming commutative squares. It is easy to
 148 see that isomorphic conditions are equivalent, but not necessarily vice versa.

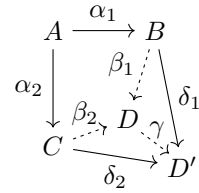
149 ► **Definition 4 (Isomorphic Conditions).** For conditions \mathcal{A}, \mathcal{B} and an iso $h: \text{RO}(\mathcal{B}) \rightarrow \text{RO}(\mathcal{A})$,
 150 we say that \mathcal{A}, \mathcal{B} are isomorphic ($\mathcal{A} \cong \mathcal{B}$) wrt. h , whenever both are universal, i.e.,
 151 $\mathcal{A} = \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i$, $\mathcal{B} = \bigwedge_{j \in J} \forall g_j. \mathcal{B}_j$, and for each $i \in I$ there exists $j \in J$ and an iso
 152 $h_{j,i}: \text{RO}(\mathcal{B}_j) \rightarrow \text{RO}(\mathcal{A}_i)$ such that $h; f_i = g_j; h_{j,i}$ and $\mathcal{A}_i \cong \mathcal{B}_j$ wrt. $h_{j,i}$; and vice versa (for
 153 each $j \in J$ there exists $i \in I \dots$). Analogously if both conditions are existential.

154 2.4 Representative Squares and the Shift Operation

155 We will now define the notion of representative squares, which describe representative ways
 156 to close a span of arrows. Such squares are intimately related to idem pushouts [15] or
 157 borrowed context diagrams [3].

158 ► **Definition 5 (Representative squares [1]).** A class κ of commuting
 159 squares in a category \mathbf{C} is representative if for every commuting square
 160 $\alpha_1; \delta_1 = \alpha_2; \delta_2$ in \mathbf{C} there exists a square $\alpha_1; \beta_1 = \alpha_2; \beta_2$ in κ and an
 161 arrow γ such that $\delta_1 = \beta_1; \gamma$ and $\delta_2 = \beta_2; \gamma$.

162 For two arrows $\alpha_1: A \rightarrow B$, $\alpha_2: A \rightarrow C$, we define $\kappa(\alpha_1, \alpha_2)$ as the
 163 set of pairs of arrows (β_1, β_2) which, together with α_1, α_2 , form representative squares in κ .



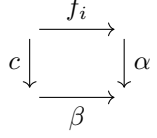
164 In categories with pushouts (such as **Graph_{fin}**), pushouts are the most natural candidate
 165 for representative squares. In **Graph_{fin}^{inj}** pushouts do not exist, but jointly epi squares can
 166 be used instead. For cospan categories, one can use borrowed context diagrams [3] (see
 167 Appendix A for a summary).

168 For many categories of interest – such as **Graph_{fin}** and **ILC(Graph_{fin})** – we can
 169 guarantee a choice of κ such that each set $\kappa(a, b)$ is finite and computable. In the rest of
 170 this paper, we assume that we work in such a category, and use such a class κ . Hence the

² Actually, without restrictions these are proper classes rather than sets. We tacitly assume that we are working in the corresponding skeleton category where no two different objects are isomorphic and assume that we can consider **Cond**, **Seq** as sets.

171 constructions described below are effective since the finiteness of the transformed conditions
 172 is preserved. One central operation is the shift of a condition along an arrow. The name
 173 shift is taken from an analogous operation for nested application conditions (see [18]).

174 ► **Definition 6** (Shift of a Condition). *Given a fixed class of representative squares κ , the*
 175 *shift of a condition \mathcal{A} along an arrow $c: A \rightarrow B$ is inductively defined as follows:*

$$176 \quad \left(\bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \right)_{\downarrow c} = \bigwedge_{i \in I} \bigwedge_{(\alpha, \beta) \in \kappa(f_i, c)} \forall \beta. (\mathcal{A}_i)_{\downarrow \alpha}$$


177 *Shifting of existential conditions is done analogously.*

178 The shift operation can be understood as a partial evaluation of \mathcal{A} under the assumption
 179 that c is already present. In particular it satisfies $[c; d_1, d_2, \dots] \models \mathcal{A} \iff [d_1, d_2, \dots] \models \mathcal{A}_{\downarrow c}$.
 180 This implies that while the representation of the shifted condition may differ depending on
 181 the chosen class of representative squares, the resulting conditions are equivalent. Since we
 182 assume that each set $\kappa(f_i, c)$ is finite, shifting a finite condition will again result in a finite
 183 condition. If one arrow is a section, the following property is useful:

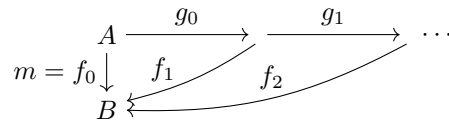
184 ► **Lemma 7** (Representative squares preserve sections). *Let $\alpha_1: A \rightarrow B$, $\alpha_2: A \rightarrow C$. If α_2 is*
 185 *a section, then there exists some $(\beta_1, \beta_2) \in \kappa(\alpha_1, \alpha_2)$ such that β_1 is a section.*

186 In the case where α_1 is an iso, we can always assume that $\kappa(\alpha_1, \alpha_2) = \{(\alpha_1^{-1}; \alpha_2, \text{id})\}$
 187 and we will use this assumption in the paper.

188 2.5 Further Concepts

189 Our goal is to develop a procedure that finds a finite model if one exists, produces unsatis-
 190 fiability proofs if a condition has neither finite nor infinite models, and otherwise does not
 191 terminate. In order to state the correctness of this procedure, we will need an abstract notion
 192 of finiteness and to this aim we introduce *finitely decomposable morphisms*. Intuitively this
 193 means that every infinite decomposition contains only finitely many non-isomorphisms.

194 ► **Definition 8** (Finitely decomposable morphism). *A morphism $m: A \rightarrow B$ is finitely decom-*
 195 *posable if for every infinite sequence of f_i, g_i , $i \in \mathbb{N}_0$, such that $f_0 = m$ and $f_i = g_i; f_{i+1}$,*
 196 *only finitely many g_i are non-isomorphisms (cf. the diagram below).*



197 Note that in $\mathbf{Graph}_{\text{fin}}^{\text{inj}}$ all arrows are finitely decomposable, while this is not the case in
 198 $\mathbf{Graph}_{\text{fin}}$. In this category there exists a section s (with associated retraction r such that
 199 $s; r = \text{id}$) that is *not* an iso. Then, the identity on the domain of s has a decomposition
 200 into infinitely many non-isos (an alternating sequence of s and r , more concretely: $g_{2i} = s$,
 201 $g_{2i+1} = r$ and $f_{2i} = \text{id}$, $f_{2i+1} = r$) and is hence not finitely decomposable.

202 While satisfaction is typically defined inductively (as in Definition 2), i.e., as a least
 203 fixpoint, we can also view it coinductively, i.e., as a greatest fixpoint, due to the fact that all
 204 conditions are finite.

205 ► **Proposition 9** (Fixpoint function for satisfaction). *Let $\bar{a} = [a_1, a_2, a_3, \dots] \in \text{Seq}$ be a*
 206 *composable sequence of arrows. We define the function $s: \mathcal{P}(\text{Seq} \times \text{Cond}) \rightarrow \mathcal{P}(\text{Seq} \times \text{Cond})$*
 207 *as follows: Let $P \subseteq \text{Seq} \times \text{Cond}$, then*

- 209 ■ $(\bar{a}, \bigwedge_i \forall f_i. \mathcal{A}_i) \in s(P)$ iff for every $i \in I$ and every arrow $g: \text{RO}(\mathcal{A}_i) \rightarrow B$ and all $n \in \mathbb{N}_0$
 210 we have: if $a_1; \dots; a_n = f_i; g$, then $([g, a_{n+1}, \dots], \mathcal{A}_i) \in P$.
 211 ■ $(\bar{a}, \bigvee_i \exists f_i. \mathcal{A}_i) \in s(P)$ iff there exists $i \in I$ and an arrow $g: \text{RO}(\mathcal{A}_i) \rightarrow B$ and some $n \in \mathbb{N}_0$
 212 such that $a_1; \dots; a_n = f_i; g$ and $([g, a_{n+1}, \dots], \mathcal{A}_i) \in P$.
 213 The least and greatest fixpoint of s ($\mu s, \nu s$) coincide and they equal the satisfaction relation \models .

3 Satisfiability Checking in the Restricted Case

215 Given a condition \mathcal{A} , we want to know whether \mathcal{A} is satisfiable and generate a finitely
 216 decomposable model, if it exists. Here we provide a procedure that works under the
 217 assumption that we are working in a category where all sections are isos. This is for instance
 218 true for $\mathbf{Graph}_{\text{fin}}^{\text{inj}}$ and $\mathbf{ILC}(\mathbf{Graph}_{\text{fin}}^{\text{inj}})$, where non-trivial right-inverses do not exist. It does
 219 not hold for left-linear cospans (counterexample: $\text{id} = \textcircled{1} \rightarrow \textcircled{1} \leftarrow \textcircled{1} \textcircled{2}; \textcircled{1} \textcircled{2} \rightarrow \textcircled{1} \textcircled{2} \leftarrow \textcircled{1}$).
 220 The general case where this assumption does not hold will be treated in Section 5.

3.1 Tableau Calculus

222 The underlying idea is fairly straightforward: we take an alternating condition \mathcal{A} and
 223 whenever it is existential, that is $\mathcal{A} = \bigvee_{i \in I} \exists f_i. \mathcal{A}_i$, we branch and check whether some \mathcal{A}_i is
 224 satisfiable. If instead it is universal, i.e., $\mathcal{A} = \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i$, we check whether some f_i is an
 225 iso. If that is not the case, clearly the sequence of identities on $\text{RO}(\mathcal{A})$ is a model, since there
 226 is no arrow g such that $\text{id} = f_i; g$, assuming that all sections are isos. If however some f_i is
 227 an iso, we invoke a pull-forward rule (more details see below) that transforms the universal
 228 condition into an existential condition and we continue from there. We will show that this
 229 procedure works whenever the pull-forward follows a *fair* strategy: in particular every iso
 230 (respectively one of its successors) must be pulled forward eventually.

231 The pull-forward rule relies on the equivalence $(\mathcal{A} \wedge \exists f. \mathcal{B}) \equiv \exists f. (\mathcal{A} \downarrow_f \wedge \mathcal{B})$ (cf. Appendix B).

232 ► **Lemma 10** (Pulling forward isomorphisms). *Let $\bigwedge_{i \in I} \forall f_i. \mathcal{A}_i$ be a universal condition and*
 233 *assume for some $p \in I$, f_p is an iso and $\mathcal{A}_p = \bigvee_{j \in J} \exists g_j. \mathcal{B}_j$. Then f_p can be pulled forward:*

$$234 \quad \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \equiv \exists f_p. \bigvee_{j \in J} \exists g_j. \left(\mathcal{B}_j \wedge \left(\bigwedge_{m \in I \setminus \{p\}} \forall f_m. \mathcal{A}_m \right) \downarrow_{f_p; g_j} \right)$$

235 ► **Definition 11** (SatCheck tableau construction rules). *Given an alternating condition \mathcal{A} , we*
 236 *give rules for the construction of a tableau for \mathcal{A} that has conditions as nodes, \mathcal{A} as root*
 237 *node, and edges labeled with arrows. The tableau is extended at its leaf nodes as follows:*

$$238 \quad \begin{array}{c|c} \text{For every } p \in I: & \text{For one } p \in I \text{ such that } f_p \text{ is iso and } \mathcal{A}_p = \bigvee_{j \in J} \exists g_j. \mathcal{B}_j: \\ \bigvee_{i \in I} \exists f_i. \mathcal{A}_i \xrightarrow{f_p} \mathcal{A}_p & \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \xrightarrow{f_p} \underbrace{\bigvee_{j \in J} \exists g_j. \left(\mathcal{B}_j \wedge \left(\bigwedge_{m \in I \setminus \{p\}} \forall f_m. \mathcal{A}_m \right) \downarrow_{f_p; g_j} \right)}_{\substack{\mathcal{A}_p \quad \text{other children, shifted to include } g_j}} \end{array}$$

- 239 ■ *For existential conditions, for each(!) child condition $\exists f_p. \mathcal{A}_p$, add a new descendant.*
 240 ■ *For universal conditions, non-deterministically pick one(!) child condition $\forall f_p. \mathcal{A}_p$ that*
 241 *can be pulled forward (f_p is an iso), pull it forward (cf. Lemma 10), and add the result as*
 242 *its (only) descendant. If a universal condition contains no isos, then add no descendant.*

243 A branch of a tableau is a (potentially infinite) path $\mathcal{A}_0 \xrightarrow{u_1} \mathcal{A}_1 \xrightarrow{e_1} \mathcal{A}_2 \xrightarrow{u_2} \dots$ starting
 244 from the root node. A finite branch is extendable if one of the tableau construction rules is

applicable at its leaf node and would result in new nodes (hence, a branch where the leaf is an empty existential or a universal without isomorphisms is not extendable). A branch is closed if it ends with an empty existential, otherwise it is open.

Due to Lemma 10 each universal condition is (up to iso f_p) equivalent to its (unique) descendant (if one exists), while an existential condition is equivalent to the disjunction of its descendants.

The labels along one branch of the tableau are arrows between the root objects of the conditions. Their composition corresponds to the prefix of a potential model being constructed step by step, which is evident especially for existential conditions. If the descendant condition is satisfiable, then its parent is also satisfiable. Finite paths represent a model if they are open and not extendable. For infinite paths, we need an additional property to make sure that the procedure does not “avoid” a possibility to show unsatisfiability of a condition.

To capture that, we introduce the notion of fairness, meaning that all parts of a condition are eventually used in a proof (a related concept is saturation, see e.g. [12]). For this we first need to track how pulling forward one child condition changes the other children by shifting. We define a *successor relation* that, for each pair of \forall -condition and one of its \forall -grandchildren, relates child conditions of the \forall -condition to their shifted counterparts in the \forall -condition of the second-next nesting level. The successor relation is similar in spirit to the one used in [12].

► **Definition 12** (Successor relation). Assume in the construction of a tableau we have a path

$$\bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \xrightarrow{f_p} \mathcal{C} \xrightarrow{g_j} \mathcal{B}_j \wedge \left(\bigwedge_{m \in I \setminus \{p\}} \forall f_m. \mathcal{A}_m \right) \downarrow_{f_p; g_j} = \mathcal{B}_j \wedge \bigwedge_{m \in I \setminus \{p\}} \bigwedge_{(\alpha, \beta) \in \kappa(f_m, f_p; g_j)} \forall \beta. (\mathcal{A}_m)_{\downarrow \alpha}$$

where \mathcal{C} is the existential condition given in Definition 11. Then for each $m \in I \setminus \{p\}$, each $\forall \beta. (\mathcal{A}_m)_{\downarrow \alpha}$ where $(\alpha, \beta) \in \kappa(f_m, f_p; g_j)$ is a successor of $\forall f_m. \mathcal{A}_m$. The transitive closure of the successor relation induces the indirect successor relation.

► **Definition 13** (Fairness). An infinite branch of a tableau is fair if for each universal condition \mathcal{A} on the branch and each child condition $\forall f_i. \mathcal{A}_i$ of \mathcal{A} where f_i is an iso, it holds that some indirect successor of $\forall f_i. \mathcal{A}_i$ is eventually pulled forward.

► **Remark 14** (Fairness strategies). One possible strategy that ensures fairness is to maintain for each incomplete branch a queue of child conditions for which a successor must be pulled forward. Then the first entry in this queue is processed. Note that by the assumption on κ made earlier at the end of Section 2.4, each iso in a universal condition that is not pulled forward has exactly one successor and the queue is modified by replacing each condition accordingly and adding newly generated child conditions with isos at the end. ◻

3.2 Up-To Techniques, Fair Branches and Models

While showing soundness of the tableau method is relatively straightforward, the crucial part of the completeness proof is to show that every infinite and fair branch of the tableau corresponds to a model. The proof strategy is the following: given such a branch, we aim to construct a witness for this model, by pairing conditions on this path with the suffix consisting of the sequence of arrows starting from this condition. If one could show that the set $P \subseteq \text{Seq} \times \text{Cond}$ of pairs so obtained is a post-fixpoint of the satisfaction function s defined in Proposition 9 ($P \subseteq s(P)$), we could conclude, as the satisfaction relation \models is the greatest fixpoint of s (Proposition 9) and hence above any post-fixpoint.

However, P is in general not a post-fixpoint, which is mainly due to the fact that universal conditions are treated “sequentially” one after another and are “pulled forward” only if they become isos. Hence, if we want to show that for a chain $[a_1, a_2, \dots]$ the universal condition of the form $\bigwedge_i \forall f_i. \mathcal{A}_i$ is satisfied, we have to prove for every child $\forall f_i. \mathcal{A}_i$ that whenever $a_1; \dots; a_n = f_i; g$ it holds that $[g, a_{n+1}, \dots] \models \mathcal{A}_i$. If $\forall f_i. \mathcal{A}_i$ actually is the child that is pulled forward in the next tableau step, P contains the tuple required by s . If not, there is a “delay” and intuitively that means that we can not guarantee that P is indeed a post-fixpoint.

However it turns out that it is a post-fixpoint up-to ($P \subseteq s(u(P))$), where u is a combination of one or more suitable up-to functions. We first explore several such up-to functions: $u_\wedge(P)$ obtains new conditions by non-deterministically removing parts of conjunctions; with $u_\S(P)$ we can arbitrarily recompose (decompose and compose) the arrows in a potential model; with $u_\downarrow(P)$ we can undo a shift; and $u_\cong(P)$ allows replacing conditions with isomorphic conditions. For each, we show their s -compatibility (i.e., $u(s(P)) \subseteq s(u(P))$).

► **Theorem 15 (Up-to techniques).** *Let $P \subseteq \text{Seq} \times \text{Cond}$, i.e. tuples of potential model and condition. Then the following four up-to functions are s -compatible:*

■ *We inductively define a relation \mathcal{U}_\wedge containing a pair of conditions $(\mathcal{A}, \mathcal{T})$ iff \mathcal{T} is the same as \mathcal{A} but with some conjunctions removed. That is, \mathcal{U}_\wedge contains*

$$\begin{array}{ll} \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i & \mathcal{U}_\wedge \quad \bigwedge_{j \in J \subseteq I} \forall f_j. \mathcal{T}_j \quad \text{whenever } (\mathcal{A}_j, \mathcal{T}_j) \in \mathcal{U}_\wedge \text{ for all } j \in J \\ \bigvee_{i \in I} \exists f_i. \mathcal{A}_i & \mathcal{U}_\wedge \quad \bigvee_{i \in I} \exists f_i. \mathcal{T}_i \quad \text{whenever } (\mathcal{A}_i, \mathcal{T}_i) \in \mathcal{U}_\wedge \text{ for all } i \in I \end{array}$$

Then define up-to conjunction removal: $u_\wedge(P) = \{(\bar{c}, \mathcal{T}) \mid (\bar{c}, \mathcal{A}) \in P, (\mathcal{A}, \mathcal{T}) \in \mathcal{U}_\wedge\}$

■ *Recomposition: $u_\S(P) = \{([b_1, \dots, b_\ell, \bar{c}], \mathcal{A}) \mid ([a_1, \dots, a_k, \bar{c}], \mathcal{A}) \in P, a_1; \dots; a_k = b_1; \dots; b_\ell\}$*

■ *Shift: $u_\downarrow(P) = \{([c; c_1, c_2, \dots], \mathcal{B}) \mid ([c_1, c_2, \dots], \mathcal{B}_{\downarrow c}) \in P\}$*

■ *Isomorphic condition: $u_\cong(P) = \{([h; c_1, c_2, \dots], \mathcal{B}) \mid ([c_1, c_2, \dots], \mathcal{A}) \in P, \mathcal{A} \cong \mathcal{B} \text{ with iso } h: \text{RO}(\mathcal{B}) \rightarrow \text{RO}(\mathcal{A})\}$*

Note however that up-to equivalence is *not* a valid up-to technique: let \mathcal{U} be an unsatisfiable condition and let $P = \{([id, \dots], \text{vid}.\mathcal{U})\}$. As $\mathcal{U} \equiv \text{vid}.\mathcal{U}$, then also $([id, \dots], \mathcal{U}) \in u_\equiv(P)$ and hence $P \subseteq s(u_\equiv(P))$. If the technique were correct, this would imply $id \models \mathcal{U}$.

A convenient property of compatibility is that it is preserved by various operations, in particular, composition, union and iteration ($f^\omega = \bigcup_{i \in \mathbb{N}_0} f^i$). This can be used to combine multiple up-to techniques into a new one that also has the compatibility property. [20]

► **Lemma 16 (combining up-to techniques).** *Let $u = (u_\S \cup u_\wedge \cup u_\downarrow \cup u_\cong)^\omega$ be the iterated application of the up-to techniques from Theorem 15, then u is s -compatible.*

We are now able to prove the central theorem needed for showing completeness.

► **Theorem 17 (Fair branches are models).** *Let \mathcal{A}_0 be an alternating condition. Let a fixed tableau constructed by the rules of Definition 11 be given. Let $\mathcal{A}_0 \xrightarrow{b_1} \mathcal{A}_1 \xrightarrow{b_2} \mathcal{A}_2 \xrightarrow{b_3} \dots$ be a branch of the tableau that is either not extendable and ends with a universal quantification (i.e., it is open), or is infinite and fair. For such a branch, we define $P = \{(\bar{b}, \mathcal{A}_i) \mid i \in \mathbb{N}_0, \bar{b} = [b_{i+1}, b_{i+2}, \dots]\} \subseteq \text{Seq} \times \text{Cond}$, i.e., the relation P pairs suffixes of the branch with the corresponding conditions. Finally, let u be the combination of up-to techniques defined in Lemma 16. Then, $P \subseteq s(u(P))$, which implies that $P \subseteq \models$. In other words, every such branch in a tableau of Definition 11 corresponds to a model of \mathcal{A}_0 .*

Proof sketch. Let $([c_1, c_2, \dots], \mathcal{C}_0) \in P$, which corresponds to a suffix $\mathcal{C}_0 \xrightarrow{c_1} \mathcal{C}_1 \xrightarrow{c_2} \mathcal{C}_2 \xrightarrow{c_3} \dots$ of the chosen branch. We show that $([c_1, c_2, \dots], \mathcal{C}_0) \in s(u(P))$:

- 329 ■ **\mathcal{C}_0 is existential:** The next label on the branch is the arrow of some child $\exists c_1.\mathcal{C}_1$ of \mathcal{C}_0
 330 and $([c_2, c_3, \dots], \mathcal{C}_1) \in P$ implies $([c_1, c_2, \dots], \mathcal{C}_0) \in s(P) \subseteq s(u(P))$.
- 331 ■ **\mathcal{C}_0 is universal and contains no iso:** The branch ends at this point, so the sequence of
 332 arrows in the tuple is empty and represents id , where $(\text{id}, \mathcal{C}_0) \in s(u(P))$: no f_i is an iso,
 333 therefore $f_i; g = \text{id}$ is never true and hence the universal condition is trivially satisfied.
- 334 ■ **\mathcal{C}_0 is universal and contains at least one iso:** By definition of s , we need to be able
 335 to satisfy any given child $\forall d_0.\mathcal{D}_0$, i.e., show that whenever $c_1; \dots; c_n = d_0; g$ for some g, n ,
 336 then $([g, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$.
 337 Fairness guarantees that an indirect successor $\forall d_q.\mathcal{D}_q$ of $\forall d_0.\mathcal{D}_0$ is pulled forward, which
 338 results (after up-to conjunction removal) in a tuple $([c_m, \dots], \mathcal{D}_q) \in u(P)$. The intermediate
 339 steps on the branch, where other children are pulled forward instead, allow expressing \mathcal{D}_q
 340 as $(\mathcal{D}_0)_{\downarrow \alpha_1 \downarrow \dots \downarrow \alpha_q}$. Use up-to shift to transform to $([\alpha_1; \dots; \alpha_q; c_m, \dots], \mathcal{D}_0) \in u(P)$, then
 341 up-to recomposition to the required $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$. ◀

3.3 Soundness and Completeness

342 We are finally ready to show soundness and completeness of our method.

343 As a condition is essentially equivalent to any of its tableaux, which break it down into
 344 existential subconditions, a closed tableau represents an unsatisfiable condition.

345 ▶ **Theorem 18 (Soundness).** *If there exists a tableau T for a condition \mathcal{A} where all branches are closed, then the condition \mathcal{A} in the root node is unsatisfiable.*

346 **Proof sketch.** By induction over the depth of T . Base case is false (obviously unsatisfiable).
 347 Induction step for $\exists: \bigvee_i \exists f_i.\mathcal{A}_i$ is unsatisfiable if all \mathcal{A}_i are. For \forall : by construction, the only
 348 child contains an equivalent condition. ◀

351 We now prove completeness, which – to a large extent – is a corollary of Theorem 17.

352 ▶ **Theorem 19 (Completeness).** *If a condition \mathcal{A} is unsatisfiable, then every tableau constructed by obeying the fairness constraint will result in a finite tableau where all branches are closed. Furthermore, at least one such tableau exists.*

353 **Proof.** The contraposition follows from Theorem 17: If the constructed tableau is finite with
 354 open branches or infinite, then \mathcal{A} is satisfiable. Furthermore, a fair tableau must exist and
 355 can be constructed by following the strategy described in Remark 14. ◀

356 In the next section we will show how the open branches in a fully expanded tableau can
 357 be interpreted as models, thus giving us a procedure for model finding.

360 ▶ **Example 20 (Proving unsatisfiability).** We now work in $\mathbf{Graph}_{\text{fin}}^{\text{inj}}$. For this example, we
 361 use the following shorthand notation for graph morphisms: $[\textcircled{1} \rightarrow \textcircled{2}]$ means $\textcircled{1} \rightarrow \textcircled{1} \rightarrow \textcircled{2}$, i.e.,
 362 the morphism is the inclusion from the light-gray graph elements to the full graph.

363 Consider the condition $\mathcal{A} = \forall[\emptyset].\exists[\textcircled{1}].\text{true} \wedge \forall[\textcircled{\otimes}].\text{false}$, meaning (1) there exists a node
 364 and (2) no node must exist. It is easily seen that these contradict each other and hence \mathcal{A} is
 365 unsatisfiable. We obtain a tableau with a single branch for this condition:

$$366 \quad \mathcal{A} \xrightarrow{[\emptyset]} \exists[\textcircled{1}].(\text{true} \wedge (\forall[\textcircled{\otimes}].\text{false})_{\downarrow[\textcircled{1}]}) \xrightarrow{[\textcircled{1}]} \forall[\textcircled{1}].\text{false} \wedge \forall[\textcircled{1} \rightarrow \textcircled{\otimes}].\text{false} \xrightarrow{[\textcircled{1}]} \text{false}$$

367 In the first step, \mathcal{A} is universal with an isomorphism $\emptyset \rightarrow \emptyset$, which is pulled forward.
 368 Together with its (only) existential child, this results in the partial model $[\textcircled{1}]$ and another
 369 universal condition with the meaning: (1) the just created node $\textcircled{1}$ must not exist, and
 370 (2) no additional node $\textcircled{\otimes}$ may exist either.

371 This condition includes another isomorphism (\mathbb{Q}) to be pulled forward. Its child
 372 $\mathcal{A}_p = \bigvee_{j \in J} \exists g_j. \mathcal{B}_j = \text{false}$ is an empty disjunction, so the tableau rule for universals adds an
 373 empty disjunction (false) as the only descendant. This closes the (only) branch, hence the
 374 initial condition \mathcal{A} can be recognized as unsatisfiable. \lrcorner

375 3.4 Model Finding

376 We will now discuss the fact that the calculus not only searches for a logical contradiction
 377 to show unsatisfiability, but at the same time tries to generate a (possibly infinite) model.
 378 We can show that *every* finitely decomposable (i.e., “finite”) model (or a prefix thereof)
 379 can be found after finitely many steps in a *fully expanded* tableau, i.e., a tableau where all
 380 branches are extended whenever possible, including infinite branches. This is a feature that
 381 distinguishes it from other known calculi for first-order logic.

382 The following lemma shows that an infinite branch always makes progress towards
 383 approximating the infinite model.

384 ► **Lemma 21.** *Let \mathcal{A} be a condition and T be a fully expanded tableau for \mathcal{A} . Then,*
 385 *for each branch it holds that it either is finite, or that there always eventually is another*
 386 *non-isomorphism on the branch.*

387 **Proof sketch.** We show that it is impossible for a path to begin with infinitely many isos.
 388 Assume we are starting with a universal condition \mathcal{A} , which results in some iso $\forall f_p. \mathcal{A}_p$ being
 389 pulled forward. Consider the children of the resulting existential condition $\bigvee_j \exists g_j. \mathcal{C}_j$. Each
 390 child where g_j is a non-iso immediately validates the statement.

391 For children where g_j is an iso, \mathcal{C}_j contains the result of shifting over an iso $f_p; g_j$. With
 392 the assumptions on κ made at the end of Section 2.4, such a shift preserves the structure of
 393 the condition and hence its “size”. Then show that $w(\mathcal{A}) > w(\mathcal{C}_j)$ for some weight function w
 394 that determines the size of a condition. As each such step reduces the weight and it cannot
 395 be reduced forever, there cannot be infinitely many leading isos on the branch. \blacktriangleleft

396 ► **Theorem 22 (Model Finding).** *Let \mathcal{A} be a condition, m a finitely decomposable arrow such*
 397 *that $m \models \mathcal{A}$ and let T be a fully expanded tableau for \mathcal{A} .*

398 *Then, there exists an open and unextendable branch with arrows c_1, \dots, c_n in T , having*
 399 *condition \mathcal{R} in the leaf node, where $m = c_1; \dots; c_n; r$ for some r with $r \models \mathcal{R}$. Furthermore*
 400 *the finite prefix is itself a model for \mathcal{A} (i.e., $[c_1, \dots, c_n] \models \mathcal{A}$).*

401 Note that this finite branch can be found in finite time, assuming a suitable strategy for
 402 exploration of the tableau such as breadth-first search or parallel processing.

403 ► **Algorithm 23 (Satisfiability Check).** *Given a condition \mathcal{A} , we define the procedure $\text{SAT}(\mathcal{A})$*
 404 *that may either produce a model $c: \text{RO}(\mathcal{A}) \rightarrow C$, answer unsat or does not terminate.*

405 ■ *Initialize the tableau with \mathcal{A} in the root node.*

406 ■ *While the tableau still has open branches:*

- 407 ■ *Select one of the open branches as the current branch, using an appropriate strategy*
 408 *that extends each open branch eventually.*
- 409 ■ *If the leaf is a universal condition without isomorphisms, terminate and return the*
 410 *labels of the current branch as model.*
- 411 ■ *Otherwise, extend the branch according to the rules of Definition 11, obeying the*
 412 *fairness constraint.*

413 ■ *If all branches are closed, terminate and answer unsat.*

This procedure has some similarities to the tableau-based reasoning from [12]. The aspect of model generation was in particular considered in [25]. Overall, we obtain the following result:

Theorem 24. *There is a one-to-one correspondence between satisfiability of a condition (\mathcal{A} unsatisfiable; \mathcal{A} has a finitely decomposable model; \mathcal{A} is satisfiable, but has no finitely decomposable model) and the output of Algorithm 23 ($SAT(\mathcal{A})$) (terminates with unsat, terminates with a model, does not terminate).*

Proof.

- \mathcal{A} unsatisfiable \iff algorithm outputs unsat: (\Rightarrow) Theorem 19, (\Leftarrow) Theorem 18
- \mathcal{A} has a finitely decomposable model \iff algorithm finds finitely decomposable model: (\Rightarrow) Theorem 22, (\Leftarrow) Theorem 17
- \mathcal{A} has only models that are not finitely decomposable \iff algorithm does not terminate: (\Rightarrow) exclusion of other possibilities for non-termination, Lemma 21 for model in the limit (\Leftarrow) Theorem 17

Example 25 (Finding finite models). We now work in $\mathbf{Graph}_{\text{fin}}^{\text{inj}}$ and use the shorthand notation introduced in Example 20. Let the following condition be given:

$$\begin{aligned} & \forall \emptyset \rightarrow \emptyset. \exists \emptyset \rightarrow \textcircled{1}. \text{true} \quad (\text{there exists a node } \textcircled{1}) \\ & \wedge \forall \emptyset \rightarrow \textcircled{1}. \exists \textcircled{1} \rightarrow \textcircled{1 \rightarrow 2}. \text{true} \quad \text{and every node has an outgoing edge to some other node} \end{aligned}$$

This condition has finite models, the smallest being the cycle $\textcircled{1} \rightarrow \textcircled{2}$. When running Algorithm 23 on this condition, it obtains the model in the following way:

1. The given condition is universal with an iso $\emptyset \rightarrow \emptyset$, which is pulled forward. Together with its (only) existential child, this results in the partial model $\textcircled{1}$ and the condition

$$\begin{aligned} & \text{true} \wedge (\forall [\textcircled{1}] . \exists [\textcircled{1 \rightarrow 2}]. \text{true})_{\downarrow [\textcircled{1}]} \\ & = \forall [\textcircled{1}] . \exists [\textcircled{1 \rightarrow 2}]. \text{true} \wedge \forall [\textcircled{1} \text{ } \textcircled{A}]. (\overbrace{\exists [\textcircled{1} \text{ } \textcircled{A} \rightarrow \textcircled{B}]. \text{true} \vee \exists [\textcircled{1} \leftarrow \textcircled{A}]. \text{true}}^{\mathcal{B}}) \end{aligned}$$

meaning: (1) the just created node $\textcircled{1}$ must have an outgoing edge; (2) and every other node A must also have an outgoing edge to either another node or to the existing node.

2. Pull forward iso $\textcircled{1}$ and extend the partial model by $\textcircled{1 \rightarrow 2}$, resulting in:

$$\begin{aligned} & \text{true} \wedge (\forall [\textcircled{1} \text{ } \textcircled{A}]. \mathcal{B})_{\downarrow [\textcircled{1 \rightarrow 2}]} = \forall [\textcircled{1 \rightarrow 2}]. \mathcal{B}_{\downarrow [\textcircled{1 \rightarrow A}]} \wedge \forall [\textcircled{1 \rightarrow 2} \text{ } \textcircled{A}]. \mathcal{B}_{\downarrow [\textcircled{1 \rightarrow 2} \text{ } \textcircled{A}]} \\ & = \forall [\textcircled{1 \rightarrow 2}]. (\exists [\textcircled{1 \rightarrow 2} \rightarrow \textcircled{3}]. \text{true} \vee \exists [\textcircled{1 \rightarrow 2} \leftarrow \textcircled{2}]. \text{true}) \\ & \wedge \forall [\textcircled{1 \rightarrow 2} \text{ } \textcircled{A}]. (\exists [\textcircled{1 \rightarrow 2} \text{ } \textcircled{A} \rightarrow \textcircled{B}]. \text{true} \vee \exists [\textcircled{1 \rightarrow 2} \leftarrow \textcircled{A}]. \text{true} \vee \exists [\textcircled{1 \rightarrow 2} \text{ } \textcircled{A}]. \text{true}) \end{aligned}$$

meaning: (1) the second node has an edge to a third node or to the first one; (2) and every other node A also has an edge to either another node or to one of the existing nodes.

3. Next, we pull forward $\textcircled{1 \rightarrow 2}$ and extend the model by $\textcircled{1 \rightarrow 2}$:

$$\text{true} \wedge (\forall [\textcircled{1 \rightarrow 2} \text{ } \textcircled{A}]. \dots)_{\downarrow [\textcircled{1 \rightarrow 2}]} = \forall [\textcircled{1 \rightarrow 2} \text{ } \textcircled{A}]. \dots$$

4. This condition does not have any children with isos, so it is satisfiable by id. This means that the composition of the partial models so far ($\textcircled{1 \rightarrow 2}$) is a model for the original condition. \lrcorner

C_0	C_1	$(C_1)_{\downarrow m}$
$\forall [\textcircled{O} \rightarrow \textcircled{1}].\text{false}$	$\forall [\textcircled{O} \rightarrow \textcircled{1} \rightarrow \textcircled{2}].\text{false}$	
$\wedge \forall [\textcircled{1} \oplus].(\exists [\textcircled{1} \oplus \rightarrow \textcircled{O}].\text{true} \vee \exists [\textcircled{1} \leftarrow \oplus].\text{true})$	$\wedge \forall [\textcircled{1} \rightarrow \textcircled{2} \oplus].(\exists [\textcircled{1} \rightarrow \textcircled{2} \oplus \rightarrow \textcircled{O}].\text{true} \vee \exists [\textcircled{1} \rightarrow \textcircled{2} \leftarrow \oplus].\text{true} \vee \exists [\oplus \rightarrow \textcircled{1} \rightarrow \textcircled{2}].\text{true})$	$\forall [\textcircled{2} \oplus].(\exists [\textcircled{2} \oplus \rightarrow \textcircled{O}].\text{true} \vee \exists [\textcircled{2} \leftarrow \oplus].\text{true})$
$\wedge \forall [\textcircled{1}].\exists [\textcircled{1} \rightarrow \textcircled{2}].\text{true}$	$\wedge \forall [\textcircled{1} \rightarrow \textcircled{2}].(\exists [\textcircled{1} \rightarrow \textcircled{2} \rightarrow \textcircled{3}].\text{true} \vee \exists [\textcircled{1} \leftarrow \textcircled{2}].\text{true})$	$\wedge \forall [\textcircled{2}].\exists [\textcircled{2} \rightarrow \textcircled{3}].\text{true}$
$\wedge \forall [\textcircled{1} \xrightarrow{\text{A}} \otimes \leftarrow \textcircled{B}].\text{false}$	$\wedge \forall [\textcircled{1} \rightarrow \textcircled{2} \xrightarrow{\text{A}} \otimes \leftarrow \textcircled{B}].\text{false}$	$\wedge \forall [\textcircled{2} \xrightarrow{\text{A}} \otimes \leftarrow \textcircled{B}].\text{false}$
$\wedge \forall [\textcircled{1} \rightarrow \otimes \leftarrow \textcircled{B}].\text{false}$	$\wedge \forall [\textcircled{1} \rightarrow \textcircled{2} \rightarrow \otimes \leftarrow \textcircled{B}].\text{false}$	$\wedge \forall [\textcircled{2} \rightarrow \otimes \leftarrow \textcircled{B}].\text{false}$
	$\wedge \forall [\textcircled{1} \rightarrow \textcircled{2} \leftarrow \textcircled{B}].\text{false}$	$\wedge \forall [\textcircled{2} \leftarrow \textcircled{B}].\text{false}$
$\wedge \dots$	$\wedge \dots$	$\wedge \dots$

■ **Table 1** Steps for the condition of Example 28, showing that a repeating infinite model exists.

context diagrams are constructed (via pushout complements). Now observe that $(C_1)_{\downarrow m}$ is similar in structure to C_0 , and in fact, using a renaming isomorphism $\iota = \textcircled{2} \rightarrow \textcircled{O} \leftarrow \textcircled{1}$, it holds that $(C_1)_{\downarrow m} \cong C_0$ wrt. ι . \lrcorner

In general this witness construction will almost never be applicable for simple graph categories, we need to work in other categories, such as cospan categories.

5 Satisfiability in the General Case

Section 3 heavily depends on the fact that all sections are isos, i.e., only isos have a right inverse. However, in the general case we might have conditions of the form $\forall s.\mathcal{A}$ where s has a right inverse r with $s;r = \text{id}$ (note that r need not be unique). This would invalidate our reasoning in the previous sections, since the identity is not necessarily a model of $\forall s.\mathcal{A}$.

► **Example 29.** We work in the category $\mathbf{Graph}_{\text{fin}}$. Consider the following condition $\mathcal{A} = \forall \textcircled{1} \rightarrow \textcircled{1} \textcircled{2} . \exists \textcircled{1} \textcircled{2} \rightarrow \textcircled{1} \rightarrow \textcircled{2} . \text{true}$, defined over a single node $\textcircled{1}$ as root object, which states that the distinguished node has an edge to every other node – including itself, since a non-injective match may merge the two nodes. The first morphism of \mathcal{A} is a section, while the second is injective, but not a section.

The identity on the single node is not a model of \mathcal{A} , but $\textcircled{1} \rightarrow \textcircled{1} \rightarrow \textcircled{1}$ is. However, the condition $\mathcal{A} \wedge \forall \textcircled{1} \rightarrow \textcircled{1} \rightarrow \textcircled{1} . \text{false}$ is unsatisfiable, a fact that would not be detected by Algorithm 23, since neither of the universal quantifiers contains an iso. \lrcorner

Hence we will now adapt the tableau calculus to deal with sections.

► **Lemma 30** (Pulling forward sections). *Let $\bigwedge_{i \in I} \forall f_i . \mathcal{A}_i$ be a universal condition and assume that f_p , $p \in I$, is a section, and r_p is a right inverse of f_p (i.e., $f_p; r_p = \text{id}$). Furthermore let $\mathcal{A}_{p \downarrow r_p} = \bigvee_{j \in J} \exists h_j . \mathcal{H}_j$ be the result of shifting the p -th child over the right inverse. Then f_p can be pulled forward:*

$$\bigwedge_{i \in I} \forall f_i . \mathcal{A}_i \equiv \bigvee_{j \in J} \exists h_j . \left(\mathcal{H}_j \wedge \left(\bigwedge_{i \in I} \forall f_i . \mathcal{A}_i \right)_{\downarrow h_j} \right)$$

As for the analogous Lemma 10, pulling forward sections produces an equivalent condition. However, here the child being pulled forward is still included in the children shifted by h_j .

XX:14 Coinductive Techniques for Checking Satisfiability of Generalized Nested Conditions

Hence the condition will increase in size, unlike for the special case. This is necessary, since f_p might have other inverses which can be used in pulling forward and might lead to new results. This leads to the following adapted rules.

► **Definition 31** (SatCheck rules, general case). *Let \mathcal{A} be an alternating condition. We can construct a tableau for \mathcal{A} by extending it at its leaf nodes as follows:*

$$\begin{array}{c|l} \text{For every } p \in I: & \text{For one } p \in I \text{ such that } f_p \text{ is section, } f_p; r_p = \text{id}, \\ & \text{and } \bigvee_{j \in J} \exists h_j. \mathcal{H}_j = \mathcal{A}_{p \downarrow r_p}: \\ \bigvee_{i \in I} \exists f_i. \mathcal{A}_i \xrightarrow{f_p} \mathcal{A}_p & \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \xrightarrow{\text{id}} \bigvee_{j \in J} \exists h_j. \left(\mathcal{H}_j \wedge \left(\bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \right)_{\downarrow h_j} \right) \end{array}$$

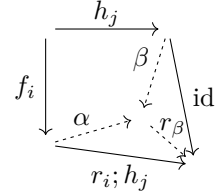
- For existential conditions, for each(!) child condition $\exists f_p. \mathcal{A}_p$, add a new descendant.
- For universal conditions, pick one(!) child condition $\forall f_p. \mathcal{A}_p$ that can be pulled forward in the sense of Lemma 30 and add the result as its (only) descendant.

The rules are similar to those of the specialized case (Definition 11) and as in the special case, we need to define a successor relation on children with sections, that in addition tracks the corresponding right-inverses. The definition of the successor relation is slightly more complex than in the previous case (Definition 12).

► **Definition 32** (Successor relation and tracking right-inverses). *We define a relation on pairs of (f_i, r_i) , where f_i is a morphism of a child of a universal condition and r_i is one of its right-inverses. Assume that in the construction of a tableau we have a path*

$$\bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \xrightarrow{\text{id}} \mathcal{C} \xrightarrow{h_j} \mathcal{H}_j \wedge \left(\bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \right)_{\downarrow h_j} = \mathcal{H}_j \wedge \bigwedge_{i \in I} \bigwedge_{(\alpha, \beta) \in \kappa(f_i, h_j)} \forall \beta. (\mathcal{A}_i)_{\downarrow \alpha}$$

where \mathcal{C} is the existential condition given in Lemma 30. Given (f_i, r_i) (where $f_i; r_i = \text{id}$), we can conclude that the outer square on the right commutes and hence there exists an inner representative square $(\alpha, \beta) \in \kappa(f_i, h_j)$ and r_β such that the diagram to the right commutes (in particular β is a section and r_β is a retraction). In this situation, we say that (β, r_β) is a (retraction) successor of (f_i, r_i) .



We extend the definition of fairness to cover sections (not just isomorphisms) and also require that all right-inverses of each section are eventually used in a pull-forward step:

► **Definition 33** (Fairness in the general case). *A branch of a tableau is fair if for each universal condition \mathcal{A} on the branch, each child condition $\forall f_i. \mathcal{A}_i$ of \mathcal{A} where f_i is a section, and each right-inverse r_i of f_i , there is $n \in \mathbb{N}_0$ such that in the n -th next step, for some indirect successor (f'_i, r'_i) of (f_i, r_i) it holds that f'_i is pulled forward using the right inverse r'_i . (Every universally-quantified section is eventually pulled forward with every inverse.)*

For this definition to be effective, we need to require that every section has only finitely many right-inverses (as in the category of finite graphs). Given that property, one way to implement a fairness strategy is to use a queue, to which child conditions (and the corresponding right-inverses) are added. This queue has to be arranged in such a way that for each section/retraction pair a successor is processed eventually.

We now show how to adapt the corresponding results of the previous section (Theorems 17–19) and in particular show that infinite and fair branches are always models, from which we can infer soundness and completeness.

► **Theorem 34** (Fair branches are models (general case)). *Let \mathcal{A}_0 be an alternating condition. Let a fixed tableau constructed by the rules of Definition 31 be given. Let $\mathcal{A}_0 \xrightarrow{b_1} \mathcal{A}_1 \xrightarrow{b_2} \mathcal{A}_2 \xrightarrow{b_3} \dots$ be a branch of the tableau that is either unextendable and ends with a universal quantification, or is infinite and fair. For such a branch, we define: $P = \{(\bar{b}, \mathcal{A}_i) \mid i \in \mathbb{N}_0, \bar{c} = [b_{i+1}, b_{i+2}, \dots]\} \subseteq \text{Seq} \times \text{Cond}$. Then, $P \subseteq s(u(P))$. (Every open and unextendable or infinite and fair branch in a tableau of Definition 31 corresponds to a model.)*

► **Theorem 35** (Soundness and Completeness).

- If all branches in a tableau are closed, then the condition in the root node is unsatisfiable.
- If a condition \mathcal{A} is unsatisfiable, then in every tableau constructed by obeying the fairness constraint all branches are closed.

Proof. Use the proof strategies of Theorems 18 and 19. Tableaux constructed by the rules of Definition 31 have all properties that are required for the proofs (in particular, universal steps lead to an equivalent condition). Use Theorem 34 to obtain models for open branches. ◀

In the general case, although soundness and completeness still hold, we are no longer able to find all finite models in finite time as before. This is intuitively due to the fact that a condition might have a finite model, but what we find is a seemingly infinite model that always has the potential to collapse to the finite model.

6 Conclusion

We have introduced satisfiability checks for nested conditions based on coinductive techniques. Naturally, there is a large choice of methods for first-order logic [5], but it is not always clear how to generalize such methods to general graph-like structures or – as done in this paper – to arbitrary categories. Our paper uses concepts from [12, 25, 16] and extends them from graphs to a generic categorical setting, stating the coinductive methods explicit.

There is a notion of Q-trees [6] reminiscent of the nested condition studied in this paper, but to our knowledge no generic satisfiability procedures have been derived for Q-trees.

Apart from tool support and optimization, future work will be directed to understanding the mechanism for witness generation in more detail. In particular, since it is known that FOL satisfiability for graphs of bounded treewidth is decidable [2], the question arises whether we can find witnesses for all models of bounded treewidth (or a suitable generalization of this notion, generalized to a categorical setting).

Furthermore we plan to transfer the technique of counterexample-guided abstraction refinement (CEGAR) [8], a program analysis technique based on abstract interpretation and predicate abstraction, to graph transformation and reactive systems. The computation of weakest preconditions and strongest postconditions for nested conditions is fairly straightforward [1] and satisfiability checks give us the necessary machinery to detect and eliminate spurious counterexamples. One still has to work around undecidability issues and understand whether there is a generalization of Craig interpolation, used to simplify predicates.

Another direction is to instantiate with a Lawvere theory [13], where arrows are n -tuples of m -ary terms. In this setting representative squares are closely related to unification.

Finally we plan to finish development of a tool that implements the satisfiability check.

References

- 1 H.J. Sander Bruggink, Raphaël Cauderlier, Mathias Hülsbusch, and Barbara König. Conditional reactive systems. In *Proc. of FSTTCS '11*, volume 13 of *LIPIcs*. Schloss Dagstuhl – Leibniz Center for Informatics, 2011.

- 598 2 Bruno Courcelle. The monadic second-order logic of graphs I. Recognizable sets of finite
599 graphs. *Information and Computation*, 85(1):12–75, 1990.
- 600 3 Hartmut Ehrig and Barbara König. Deriving bisimulation congruences in the DPO approach
601 to graph rewriting with borrowed contexts. *MSCS*, 16(6):1133–1163, 2006.
- 602 4 Hartmut Ehrig, Michael Pfender, and Hans Jürgen Schneider. Graph grammars: An algebraic
603 approach. In *Proc. 14th IEEE Symp. on Switching and Automata Theory*, pages 167–180,
604 1973.
- 605 5 Melvin Fitting. *First-Order Logic and Automated Theorem Proving*. Springer, 1996.
- 606 6 Peter J. Freyd and Andre Scedrov. *Categories, Allegories*. North-Holland, 1990.
- 607 7 Annegret Habel and Karl-Heinz Pennemann. Correctness of high-level transformation systems
608 relative to nested conditions. *Mathematical Structures in Computer Science*, 19(2):245–296,
609 2009.
- 610 8 Thomas A. Henzinger, Ranjit Jhala, Rupak Majumdar, and Kenneth L. McMillan. Abstractions
611 from proofs. In *Proc. of POPL '04*, pages 232–244. ACM, 2004.
- 612 9 Mathias Hülsbusch and Barbara König. Deriving bisimulation congruences for conditional
613 reactive systems. In *Proc. of FOSSACS '12*, pages 361–375. Springer, 2012. LNCS/ARCoSS
614 7213.
- 615 10 Mathias Hülsbusch, Barbara König, Sebastian Küpper, and Lara Stoltzenow. Conditional
616 Bisimilarity for Reactive Systems. *Logical Methods in Computer Science*, Volume 18, Issue 1,
617 January 2022.
- 618 11 Stephen Lack and Paweł Sobociński. Adhesive and quasiadhesive categories. *RAIRO –*
619 *Theoretical Informatics and Applications*, 39(3), 2005.
- 620 12 Leen Lambers and Fernando Orejas. Tableau-based reasoning for graph properties. In *Proc.*
621 *of ICGT '14*, pages 17–32. Springer, 2014. LNCS 8571.
- 622 13 William Lawvere. *Functorial Semantics of Algebraic Theories*. PhD thesis, Columbia University,
623 1963.
- 624 14 James J. Leifer. *Operational congruences for reactive systems*. PhD thesis, University of
625 Cambridge Computer Laboratory, September 2001.
- 626 15 James J. Leifer and Robin Milner. Deriving bisimulation congruences for reactive systems. In
627 *Proc. of CONCUR 2000*, pages 243–258. Springer, 2000. LNCS 1877.
- 628 16 Marisa Navarro, Fernando Orejas, Elvira Pino, and Leen Lambers. A navigational logic for
629 reasoning about graph properties. *Journal of Logical and Algebraic Methods in Programming*,
630 118:100616, 2021.
- 631 17 Karl-Heinz Pennemann. An algorithm for approximating the satisfiability problem of high-level
632 conditions. In *GT-VC@CONCUR*, volume 213.1 of *Electronic Notes in Theoretical Computer*
633 *Science*, pages 75–94. Elsevier, 2007.
- 634 18 Karl-Heinz Pennemann. *Development of Correct Graph Transformation Systems*. PhD thesis,
635 Universität Oldenburg, May 2009.
- 636 19 Damien Pous. Complete lattices and up-to techniques. In *Proc. of APLAS '07*, pages 351–366.
637 Springer, 2007. LNCS 4807.
- 638 20 Damien Pous. *Techniques modulo pour les bisimulations*. PhD thesis, ENS Lyon, February
639 2008.
- 640 21 Damien Pous and Davide Sangiorgi. Enhancements of the coinductive proof method. In
641 Davide Sangiorgi and Jan Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*.
642 Cambridge University Press, 2011.
- 643 22 Arend Rensink. Representing first-order logic using graphs. In *Proc. of ICGT '04*, pages
644 319–335. Springer, 2004. LNCS 3256.
- 645 23 Davide Sangiorgi. *Introduction to Bisimulation and Coinduction*. Cambridge University Press,
646 2011.
- 647 24 Vladimiro Sassone and Paweł Sobociński. Reactive systems over cospans. In *Proc. of LICS*
648 *'05*, pages 311–320. IEEE, 2005.

- 649 25 Sven Schneider, Leen Lambers, and Fernando Orejas. Symbolic model generation for graph
650 properties. In *Proc. of FASE '17*, pages 226–243. Springer, 2017. LNCS 10202.
- 651 26 Paweł Sobociński. *Deriving process congruences from reaction rules*. PhD thesis, Department
652 of Computer Science, University of Aarhus, 2004.
- 653 27 Alfred Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of*
654 *Mathematics*, 5:285–309, 1955.

655 A Proofs and Additional Material for §2 (Preliminaries)

656 Graphs and graph morphisms

657 We will define in more detail which graphs and graph morphisms we are using: in particular, a
 658 graph is a tuple $G = (V, E, s, t, \ell)$, where V, E are sets of nodes respectively edges, $s, t: E \rightarrow V$
 659 are the source and target functions and $\ell: V \rightarrow \Lambda$ (where Λ is a set of labels) is the node
 660 labelling function. In the examples we will always omit node labels by assuming that there
 661 is only a single label.

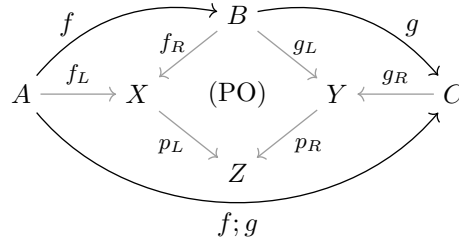
662 A graph G is finite if both V and E are finite.

663 Furthermore, given two graphs $G_i = (V_i, E_i, s_i, t_i, \ell_i)$, $i \in \{1, 2\}$, a graph morphism
 664 $\varphi: G_1 \rightarrow G_2$ consists of two maps $\varphi_V: V_1 \rightarrow V_2$, $\varphi_E: E_1 \rightarrow E_2$ such that $\varphi_V \circ s_1 = s_2 \circ \varphi_E$,
 665 $\varphi_V \circ t_1 = t_2 \circ \varphi_E$ and $\ell_1 = \ell_2 \circ \varphi_V$.

666 In the examples, the mapping of a morphism is given implicitly by the node identifiers:
 667 for instance, $\textcircled{1} \textcircled{2} \rightarrow \textcircled{1} \textcircled{3} \textcircled{2}$ adds the node identified by 3 and adds two edges from the
 668 existing nodes identified by 1 and 2.

669 Cospans and cospan composition

670 Two cospans $f: A \xrightarrow{f_L} X \xleftarrow{f_R} B$, $g: B \xrightarrow{g_L} Y \xleftarrow{g_R} C$ are composed by taking the pushout
 671 (p_L, p_R) of (f_R, g_L) as shown in Figure 1. The result is the cospan $f; g: A \xrightarrow{f_L; p_L} Z \xleftarrow{g_R; p_R} C$,
 672 where Z is the pushout object of f_R, g_L . We see an arrow $f: A \rightarrow C$ of $\mathbf{Cospans}(\mathbf{D})$ as an
 673 object B of \mathbf{D} equipped with two interfaces A, C and corresponding arrows f_L, f_R to relate
 674 the interfaces to B , and composition glues the inner objects of two cospans via their common
 675 interface.



■ **Figure 1** Composition of cospans f and g is done via pushouts

676 In order to make sure that arrow composition in $\mathbf{Cospans}(\mathbf{D})$ is associative on the nose,
 677 we quotient cospans up to isomorphism. In more detail: two cospans $f: A \xrightarrow{f_L} X \xleftarrow{f_R} B$,
 678 $g: A \xrightarrow{g_L} Y \xleftarrow{g_R} B$ are equivalent whenever there exists an iso $\iota: X \rightarrow Y$ such that $f_L; \iota = g_L$,
 679 $f_R; \iota = g_R$. Then, arrows are equivalence classes of cospans.

680 Equivalence laws for conditions

681 We rely on the results given in the following two propositions that were shown in [1]. They
 682 were originally stated for satisfaction with single arrows, but it is easy to see they are valid
 683 for possibly infinite sequences as well: conditions have finite depth and satisfaction only
 684 refers to finite prefixes of the sequence.

685 ► **Proposition 36 (Adjunction).** *Let \mathcal{A}, \mathcal{B} be two conditions with root object A , let \mathcal{C}, \mathcal{D} be*
 686 *two conditions with root object B and let $\varphi: A \rightarrow B$. Then it holds that:*

- 687 1. $\mathcal{A} \models \mathcal{B}$ implies $\mathcal{A}_{\downarrow\varphi} \models \mathcal{B}_{\downarrow\varphi}$.
- 688 2. $\mathcal{C} \models \mathcal{D}$ implies $\mathcal{Q}\varphi.\mathcal{C} \models \mathcal{Q}\varphi.\mathcal{D}$ for $\mathcal{Q} \in \{\exists, \forall\}$.
- 689 3. $\exists\varphi.(\mathcal{A}_{\downarrow\varphi}) \models \mathcal{A}$ and for every \mathcal{C} with $\exists\varphi.\mathcal{C} \models \mathcal{A}$ we have that $\mathcal{C} \models \mathcal{A}_{\downarrow\varphi}$.
- 690 4. $\mathcal{A} \models \forall\varphi.(\mathcal{A}_{\downarrow\varphi})$ and for every \mathcal{C} with $\mathcal{A} \models \forall\varphi.\mathcal{C}$ we have that $\mathcal{A}_{\downarrow\varphi} \models \mathcal{C}$.

691 ► **Proposition 37** (Laws for conditions). *One easily obtains the following laws for shift and*
 692 *quantification, conjunction and disjunction:*

$$\begin{array}{ll}
 693 \quad \mathcal{A}_{\downarrow\text{id}} \equiv \mathcal{A} & \mathcal{A}_{\downarrow\varphi;\psi} \equiv (\mathcal{A}_{\downarrow\varphi})_{\downarrow\psi} \\
 694 \quad \forall\text{id}.\mathcal{A} \equiv \mathcal{A} & \forall(\varphi;\psi).\mathcal{A} \equiv \forall\varphi.\forall\psi.\mathcal{A} \\
 695 \quad \exists\text{id}.\mathcal{A} \equiv \mathcal{A} & \exists(\varphi;\psi).\mathcal{A} \equiv \exists\varphi.\exists\psi.\mathcal{A} \\
 696 \quad (\mathcal{A} \wedge \mathcal{B})_{\downarrow\varphi} \equiv \mathcal{A}_{\downarrow\varphi} \wedge \mathcal{B}_{\downarrow\varphi} & (\mathcal{A} \vee \mathcal{B})_{\downarrow\varphi} \equiv \mathcal{A}_{\downarrow\varphi} \vee \mathcal{B}_{\downarrow\varphi} \\
 697 \quad \forall\varphi.(\mathcal{A} \wedge \mathcal{B}) \equiv \forall\varphi.\mathcal{A} \wedge \forall\varphi.\mathcal{B} & \exists\varphi.(\mathcal{A} \vee \mathcal{B}) \equiv \exists\varphi.\mathcal{A} \vee \exists\varphi.\mathcal{B}
 \end{array}$$

698 Borrowed context diagrams

699 For cospan categories over adhesive categories (such as $\mathbf{ILC}(\mathbf{Graph}_{\text{fin}})$), borrowed context
 700 diagrams – initially introduced as an extension of DPO rewriting [3] – can be used as
 701 representative squares. Before we can introduce such diagrams, we first need the notion of
 702 jointly epi.

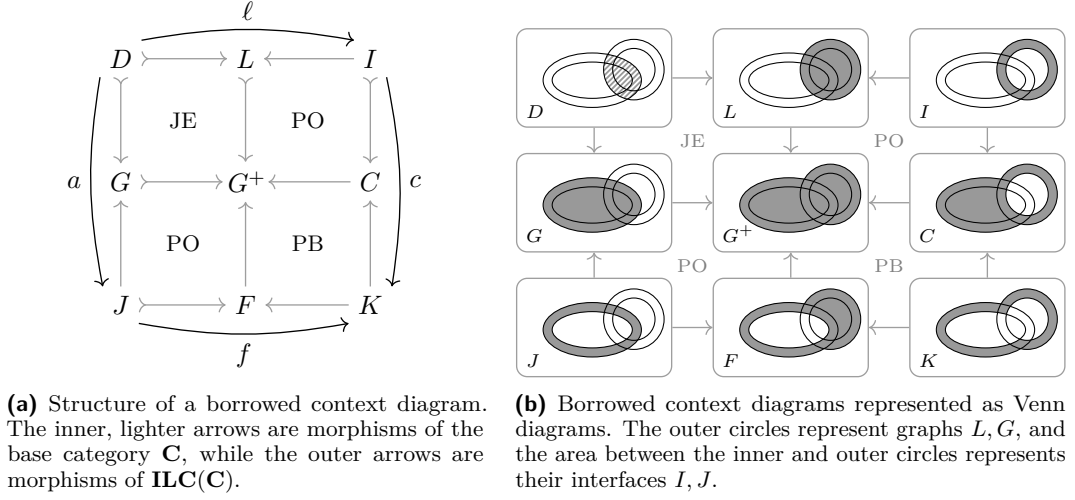
703 ► **Definition 38** (Jointly epi). *A pair of arrows $f: B \rightarrow D$, $g: C \rightarrow D$ is jointly epi (JE) if*
 704 *for each pair of arrows $d_1, d_2: D \rightarrow E$ the following holds: if $f; d_1 = f; d_2$ and $g; d_1 = g; d_2$,*
 705 *then $d_1 = d_2$.*

706 In $\mathbf{Graph}_{\text{fin}}$ jointly epi equals jointly surjective, meaning that each node or edge of D is
 707 required to have a preimage under f or g or both (it contains only images of B or C).

708 This criterion is similar to, but weaker than a pushout: For jointly epi graph morphisms
 709 $d_1: B \rightarrow D$, $d_2: C \rightarrow D$, there are no restrictions on which elements of B, C can be merged
 710 in D . However, in a pushout constructed from morphisms $a_1: A \rightarrow B$, $a_2: A \rightarrow C$, elements
 711 in D can (and must) only be merged if they have a common preimage in A . (Hence every
 712 pushout generates a pair of jointly epi arrows, but not vice versa.)

713 ► **Definition 39** (Borrowed context diagram [9]). *A commuting diagram in the category*
 714 *$\mathbf{ILC}(\mathbf{C})$, where \mathbf{C} is adhesive, is a borrowed context diagram whenever it has the form of*
 715 *the diagram shown in Figure 2a, and the four squares in the base category \mathbf{C} are pushout*
 716 *(PO), pullback (PB) or jointly epi (JE) as indicated. Arrows depicted as \rightarrow are mono. In*
 717 *particular $L \rightarrow G^+$, $G \rightarrow G^+$ must be jointly epi.*

718 Figure 2b shows a more concrete version of Figure 2a, where graphs and their overlaps
 719 are depicted by Venn diagrams (assuming that all morphisms are injective). Because
 720 of the two pushout squares, this diagram can be interpreted as composition of cospans
 721 $a; f = \ell; c = D \rightarrow G^+ \leftarrow K$ with extra conditions on the top left and the bottom right
 722 square. The top left square fixes an overlap G^+ of L and G , while D is contained in the
 723 intersection of L and G (shown as a hatched area). Being jointly epi ensures that it really is
 724 an overlap and does not contain unrelated elements. The top right pushout corresponds to
 725 the left pushout of a DPO rewriting diagram. It contains a total match of L in G^+ . Then,
 726 the bottom left pushout gives us the minimal borrowed context F such that applying the
 727 rule becomes possible. The bottom right pullback ensures that the interface K is as large as
 728 possible.



■ **Figure 2** Borrowed context diagrams

For more concrete examples of borrowed context diagrams, we refer to [3, 10].

For cospan categories over adhesive categories, borrowed context diagrams form a representative class of squares [1]. Furthermore, for some categories (such as $\mathbf{Graph}_{\mathbf{fin}}^{\text{inj}}$), there are – up to isomorphism – only finitely many jointly epi squares for a given span of monos and hence only finitely many borrowed context diagrams given a, ℓ (since pushout complements along monos in adhesive categories are unique up to isomorphism).

Whenever the two cospans ℓ, a are in $\mathbf{ILC}(\mathbf{Graph}_{\mathbf{fin}}^{\text{inj}})$, it is easy to see that f, c are in $\mathbf{ILC}(\mathbf{Graph}_{\mathbf{fin}}^{\text{inj}})$, i.e., they consist only of monos, i.e., injective morphisms.

Note also that representative squares in $\mathbf{Graph}_{\mathbf{fin}}^{\text{inj}}$ are simply jointly epi squares and they can be straightforwardly extended to squares of $\mathbf{ILC}(\mathbf{Graph}_{\mathbf{fin}}^{\text{inj}})$.

Visualization of shifts

Given a condition \mathcal{A} and an arrow $c: A = \text{RO}(\mathcal{A}) \rightarrow B$, we will visualize shifts in diagrams as follows:

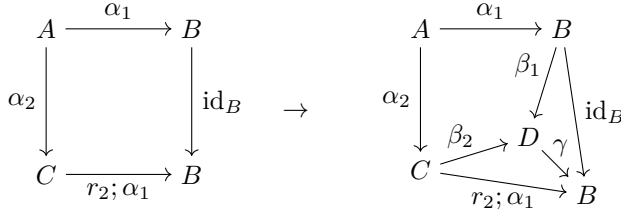
$$\begin{array}{ccc} \mathcal{A} & & \mathcal{A}_{\downarrow c} \\ \triangleright & & \triangleright \\ A & \xrightarrow{c} & B \end{array} \quad \begin{array}{ccc} & & \\ & & \\ B & \xrightarrow{d} & X \end{array}$$

Remember that for an arrow $d: B \rightarrow X$ it holds that $d \models \mathcal{A}_{\downarrow c} \iff c; d \models \mathcal{A}$.

Proofs

► **Lemma 7** (Representative squares preserve sections). *Let $\alpha_1: A \rightarrow B$, $\alpha_2: A \rightarrow C$. If α_2 is a section, then there exists some $(\beta_1, \beta_2) \in \kappa(\alpha_1, \alpha_2)$ such that β_1 is a section.*

Proof. Consider the commuting square given below on the left, where r_2 is some right-inverse of α_2 , i.e., $\alpha_2; r_2 = \text{id}$. Since every commuting square can be reduced to a representative square, there exist β_1, β_2, γ as given in the diagram below on the right. Since $\beta_1; \gamma = \text{id}_B$ is an isomorphism, β_1 is a section and γ a retraction.



751

752 ► **Proposition 9** (Fixpoint function for satisfaction). Let $\bar{a} = [a_1, a_2, a_3, \dots] \in \text{Seq}$ be a
 753 composable sequence of arrows. We define the function $s: \mathcal{P}(\text{Seq} \times \text{Cond}) \rightarrow \mathcal{P}(\text{Seq} \times \text{Cond})$
 754 as follows: Let $P \subseteq \text{Seq} \times \text{Cond}$, then

- 755 ■ $(\bar{a}, \bigwedge_i \forall f_i. \mathcal{A}_i) \in s(P)$ iff for every $i \in I$ and every arrow $g: \text{RO}(\mathcal{A}_i) \rightarrow B$ and all $n \in \mathbb{N}_0$
 756 we have: if $a_1; \dots; a_n = f_i; g$, then $([g, a_{n+1}, \dots], \mathcal{A}_i) \in P$.
- 757 ■ $(\bar{a}, \bigvee_i \exists f_i. \mathcal{A}_i) \in s(P)$ iff there exists $i \in I$ and an arrow $g: \text{RO}(\mathcal{A}_i) \rightarrow B$ and some $n \in \mathbb{N}_0$
 758 such that $a_1; \dots; a_n = f_i; g$ and $([g, a_{n+1}, \dots], \mathcal{A}_i) \in P$.

759 The least and greatest fixpoint of s ($\mu s, \nu s$) coincide and they equal the satisfaction relation \models .

760 **Proof.** First note that Definition 2 is seen inductively the least fixpoint spelled out there is
 761 exactly the least fixpoint of s , hence $\models = \mu s$.

762 It also holds that $\nu s = \models$ (the greatest fixpoint of s equals the satisfaction relation).
 763 The correctness of this characterization (i.e. that “ s is the right function”) can be shown as
 764 follows: $\models \subseteq \nu s$ because \models is a fixpoint of s . To show $\nu s \subseteq \models$, let $(\bar{c}, \mathcal{A}) \in \nu s = s(\nu s)$ (the
 765 latter holds because νs is a fixpoint). We show $(\bar{c}, \mathcal{A}) \in \models$ by induction on the nesting depth
 766 of \mathcal{A} :

- 767 ■ Let the depth be 1, i.e., \mathcal{A} has no children. Then it must be an empty universal
 768 quantification, i.e., true. Hence also $\bar{c} \models \mathcal{A}$.
- 769 ■ Let the statement hold for depth d and let \mathcal{A} have depth $d + 1$. Let $(\bar{c}, \mathcal{A}) \in s(\nu s)$.
 770 If $\mathcal{A} = \bigvee_i \exists f_i. \mathcal{A}_i$ is existential, there exists n, i, g such that $c_1; \dots; c_n = f_i; g$ and
 771 $([g, c_{n+1}, \dots], \mathcal{A}_i) \in \nu s$. \mathcal{A}_i has depth at most d , so by the induction hypothesis also
 772 $[g, c_{n+1}, \dots] \models \mathcal{A}_i$. Then also $\bar{c} \models \mathcal{A}$. The universal case $\mathcal{A} = \bigwedge_i \forall f_i. \mathcal{A}_i$ can be shown
 773 analogously. ◀

774 **B** Proofs and Additional Material for §3 (Satisfiability Checking in the 775 Restricted Case)

776 ► **Lemma 40.** Let \mathcal{A}, \mathcal{B} be two conditions with root objects A, B and let $f: A \rightarrow B$. Then

$$777 (\mathcal{A} \wedge \exists f. \mathcal{B}) \equiv \exists f. (\mathcal{A}_{\downarrow f} \wedge \mathcal{B}) \quad (\mathcal{A} \vee \forall f. \mathcal{B}) \equiv \forall f. (\mathcal{A}_{\downarrow f} \vee \mathcal{B})$$

778 **Proof.** For $(\mathcal{A} \wedge \exists f. \mathcal{B}) \equiv \exists f. (\mathcal{A}_{\downarrow f} \wedge \mathcal{B})$: let c be an arrow with source object $\text{RO}(\mathcal{A})$. Then:

$$\begin{aligned}
 779 c \models (\mathcal{A} \wedge \exists f. \mathcal{B}) &\iff c \models \mathcal{A} \wedge c \models \exists f. \mathcal{B} \\
 780 (\text{Definition 2}) &\iff c \models \mathcal{A} \wedge \exists \alpha (c = f; \alpha \wedge \alpha \models \mathcal{B}) \\
 781 &\iff \exists \alpha (c = f; \alpha \wedge c \models \mathcal{A} \wedge \alpha \models \mathcal{B}) \\
 782 &\iff \exists \alpha (c = f; \alpha \wedge f; \alpha \models \mathcal{A} \wedge \alpha \models \mathcal{B}) \\
 783 (\text{Definition 6}) &\iff \exists \alpha (c = f; \alpha \wedge \alpha \models \mathcal{A}_{\downarrow f} \wedge \alpha \models \mathcal{B}) \\
 784 &\iff \exists \alpha (c = f; \alpha \wedge \alpha \models (\mathcal{A}_{\downarrow f} \wedge \mathcal{B})) \\
 785 (\text{Definition 2}) &\iff c \models \exists f. (\mathcal{A}_{\downarrow f} \wedge \mathcal{B})
 \end{aligned}$$

786 The dual equivalence can be derived from the first one by negation. ◀

787 ► **Lemma 41.** *If f is an isomorphism, then $\forall f.\mathcal{A} \equiv \exists f.\mathcal{A}$.*

Proof.

$$\begin{aligned}
 & c \models \forall f.\mathcal{A} \\
 (\text{def sat}) & \iff \forall \alpha: \text{if } c = f; \alpha \text{ then } \alpha \models \mathcal{A} \\
 (f \text{ iso}) & \iff \forall \alpha: \text{if } f^{-1}; c = f^{-1}; f; \alpha \text{ then } \alpha \models \mathcal{A} \\
 & \iff \forall \alpha: \text{if } f^{-1}; c = \alpha \text{ then } \alpha \models \mathcal{A} \\
 & \iff \forall \alpha: \text{if } f^{-1}; c = \alpha \text{ then } f^{-1}; c \models \mathcal{A} \\
 788 & \iff f^{-1}; c \models \mathcal{A} \\
 (\text{existential introduction}) & \iff \exists \alpha: f^{-1}; c = \alpha \text{ and } f^{-1}; c \models \mathcal{A} \\
 & \iff \exists \alpha: f^{-1}; c = \alpha \text{ and } \alpha \models \mathcal{A} \\
 (f \text{ iso}) & \iff \exists \alpha: f; f^{-1}; c = f; \alpha \text{ and } \alpha \models \mathcal{A} \\
 & \iff \exists \alpha: c = f; \alpha \text{ and } \alpha \models \mathcal{A} \\
 (\text{def sat}) & \iff c \models \exists f.\mathcal{A}
 \end{aligned}$$

789

790 ► **Lemma 10** (Pulling forward isomorphisms). *Let $\bigwedge_{i \in I} \forall f_i.\mathcal{A}_i$ be a universal condition and*
 791 *assume for some $p \in I$, f_p is an iso and $\mathcal{A}_p = \bigvee_{j \in J} \exists g_j.\mathcal{B}_j$. Then f_p can be pulled forward:*

$$792 \quad \bigwedge_{i \in I} \forall f_i.\mathcal{A}_i \equiv \exists f_p. \bigvee_{j \in J} \exists g_j. \left(\mathcal{B}_j \wedge \left(\bigwedge_{m \in I \setminus \{p\}} \forall f_m.\mathcal{A}_m \right) \downarrow_{f_p; g_j} \right)$$

793 **Proof.** The following calculations rely on Proposition 37.

$$\begin{aligned}
 & \left(\bigwedge_{i \in I} \forall f_i.\mathcal{A}_i \right) = \forall f_p.\mathcal{A}_p \quad \wedge \quad \left(\bigwedge_{m \in I \setminus \{p\}} \forall f_m.\mathcal{A}_m \right) \\
 & = \left(\forall f_p. \left(\bigvee_{j \in J} \exists g_j.\mathcal{B}_j \right) \right) \wedge \left(\bigwedge_{m \in I \setminus \{p\}} \forall f_m.\mathcal{A}_m \right) \\
 & (\text{Lemma 41}) \equiv \underbrace{\left(\exists f_p. \left(\bigvee_{j \in J} \exists g_j.\mathcal{B}_j \right) \right)}_{\substack{\exists f.\mathcal{B} \\ \exists f.(\mathcal{B})}} \wedge \underbrace{\left(\bigwedge_{m \in I \setminus \{p\}} \forall f_m.\mathcal{A}_m \right)}_{\substack{\wedge \mathcal{A} \\ \wedge \mathcal{A}_{\downarrow f}}} \\
 794 & (\text{Lemma 40}) \equiv \exists f_p. \left(\bigvee_{j \in J} \exists g_j.\mathcal{B}_j \quad \wedge \quad \left(\bigwedge_{m \in I \setminus \{p\}} (\forall f_m.\mathcal{A}_m) \right) \downarrow_{f_p} \right) \\
 & (\text{distributivity}) \equiv \exists f_p. \bigvee_{j \in J} \left(\exists g_j.\mathcal{B}_j \quad \wedge \quad \bigwedge_{m \in I \setminus \{p\}} (\forall f_m.\mathcal{A}_m) \downarrow_{f_p} \right) \\
 & (\text{Lemma 40}) \equiv \exists f_p. \bigvee_{j \in J} \exists g_j. \left(\mathcal{B}_j \quad \wedge \quad \bigwedge_{m \in I \setminus \{p\}} (\forall f_m.\mathcal{A}_m) \downarrow_{f_p; g_j} \right) \\
 & \equiv \exists f_p. \bigvee_{j \in J} \exists g_j. \left(\mathcal{B}_j \quad \wedge \quad \bigwedge_{m \in I \setminus \{p\}} (\forall f_m.\mathcal{A}_m) \downarrow_{f_p; g_j} \right)
 \end{aligned}$$

795 Note that if $(\bigwedge_{i \in I} \forall f_i.\mathcal{A}_i)$ is alternating, then so is the condition after $\exists f_p$ in the last line
 796 above. ◀

797 ► **Lemma 42** (Up-to conjunction removal). *Let $P \subseteq \text{Seq} \times \text{Cond}$, i.e. a set of tuples of potential*
 798 *model and condition. We inductively define a relation \mathcal{U}_\wedge containing a pair of conditions*
 799 *$(\mathcal{A}, \mathcal{T})$ iff \mathcal{T} is the same as \mathcal{A} but with some conjunctions removed. That is, \mathcal{U}_\wedge contains*

$$\begin{aligned}
 & \bigwedge_{i \in I} \forall f_i.\mathcal{A}_i \quad \mathcal{U}_\wedge \quad \bigwedge_{j \in J \subseteq I} \forall f_j.\mathcal{T}_j \quad \text{whenever } (\mathcal{A}_j, \mathcal{T}_j) \in \mathcal{U}_\wedge \text{ for all } j \in J \\
 800 & \bigvee_{i \in I} \exists f_i.\mathcal{A}_i \quad \mathcal{U}_\wedge \quad \bigvee_{i \in I} \exists f_i.\mathcal{T}_i \quad \text{whenever } (\mathcal{A}_i, \mathcal{T}_i) \in \mathcal{U}_\wedge \text{ for all } i \in I
 \end{aligned}$$

Also, define $u_{\wedge}(P) = \{(\bar{c}, \mathcal{T}) \mid (\bar{c}, \mathcal{A}) \in P, (\mathcal{A}, \mathcal{T}) \in \mathcal{U}_{\wedge}\}$. The function u_{\wedge} is s -compatible.

Proof. We show that $(\bar{c}, \mathcal{T}) \in u_{\wedge}(s(P))$ implies $(\bar{c}, \mathcal{T}) \in s(u_{\wedge}(P))$.

Let $(\bar{c}, \mathcal{T}) \in u_{\wedge}(s(P))$. \mathcal{T} is obtained from some \mathcal{A} by forgetting elements of a conjunction, i.e., $(\mathcal{A}, \mathcal{T}) \in \mathcal{U}_{\wedge}$, and $(\bar{c}, \mathcal{A}) \in s(P)$. There are two cases:

■ **($\mathcal{T} = \bigvee_{i \in I} \exists f_i. \mathcal{T}_i$, i.e., existential):**

In this case we know that $\mathcal{A} = \bigvee_{i \in I} \exists f_i. \mathcal{A}_i$ and $(\mathcal{A}_i, \mathcal{T}_i) \in \mathcal{U}_{\wedge}$ for all $i \in I$.

Since $(\bar{c}, \mathcal{A}) \in s(P)$, there exists some $i \in I$ and some g, n such that $c_1; \dots; c_n = f_i; g$ and $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$. We know that $(\mathcal{A}_i, \mathcal{T}_i) \in \mathcal{U}_{\wedge}$, which implies that $([g, c_{n+1}, \dots], \mathcal{T}_i) \in u_{\wedge}(P)$.

This however is exactly what is required for $(\bar{c}, \mathcal{T}) \in s(u_{\wedge}(P))$ to hold, according to the definition of s .

■ **($\mathcal{T} = \bigwedge_{j \in J} \forall f_j. \mathcal{T}_j$, i.e., universal):**

In this case we know that $\mathcal{A} = \bigwedge_{j \in J} \forall f_j. \mathcal{A}_i$ and $(\mathcal{A}_i, \mathcal{T}_i) \in \mathcal{U}_{\wedge}$ for all $i \in J$.

Since $(\bar{c}, \mathcal{A}) \in s(P)$, for all $i \in I$ and all g, n , it holds that if $c_1; \dots; c_n = f_i; g$, then $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$. In these situations, from $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$ and $(\mathcal{A}_i, \mathcal{T}_i)$ (for $i \in J \subseteq I$), it follows that $([g, c_{n+1}, \dots], \mathcal{T}_i) \in u_{\wedge}(P)$ for all $i \in J$.

This however is exactly what is required for $(\bar{c}, \mathcal{T}) \in s(u_{\wedge}(P))$ to hold, according to the definition of s . ◀

► **Lemma 43 (Up-to recomposition).** Let $P \subseteq \text{Seq} \times \text{Cond}$. Then, $u_{\S}(P)$ allows splitting and merging parts of the sequence of arrows, i.e., $u_{\S}(P) = \{([b_1, \dots, b_{\ell}, \bar{c}], \mathcal{A}) \mid ([a_1, \dots, a_k, \bar{c}], \mathcal{A}) \in P, a_1; \dots; a_k = b_1; \dots; b_{\ell}\}$. The function u_{\S} is s -compatible.

Proof. We show that $([b_1, \dots, b_{\ell}, \bar{c}], \mathcal{A}) \in u_{\S}(s(P))$ implies $([b_1, \dots, b_{\ell}, \bar{c}], \mathcal{A}) \in s(u_{\S}(P))$.

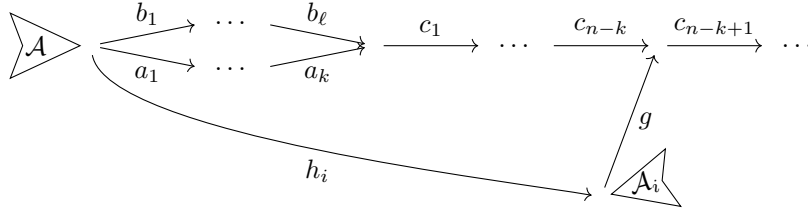
Since $([b_1, \dots, b_{\ell}, \bar{c}], \mathcal{A}) \in u_{\S}(s(P))$, there must be some $([a_1, \dots, a_k, \bar{c}], \mathcal{A}) \in s(P)$ such that $a_1; \dots; a_k = b_1; \dots; b_{\ell}$. Furthermore, let $\bar{c} = [c_1, c_2, \dots]$.

To show that $([b_1, \dots, b_{\ell}, \bar{c}], \mathcal{A}) \in s(u_{\S}(P))$, we consider the following cases:

■ **(\mathcal{A} is existential (\exists)):**

Since $([a_1, \dots, a_k, \bar{c}], \mathcal{A}) \in s(P)$ (*), there exists some child $\exists h_i. \mathcal{A}_i$ of \mathcal{A} , some g and n that satisfy the conditions of s . We consider the following subcases, depending on the value of n given:

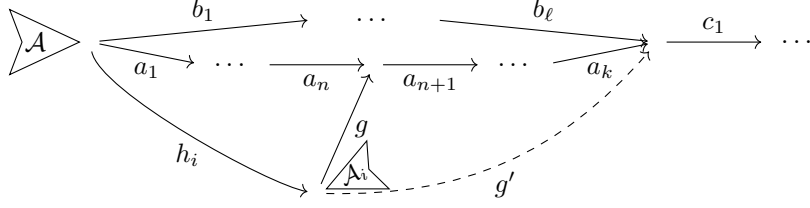
■ **($n \geq k$, i.e., $h_i; g$ “points to somewhere” in \bar{c}):**



Then we set $g = g'$ and $n' = n + \ell - k$ is the length of the prefix of $[b_1, \dots, b_{\ell}, \bar{c}]$ that is factored. It holds that:

1. $h_i; g' = h_i; g \stackrel{(1)}{=} a_1; \dots; a_k; c_1; \dots; c_{n-k} \stackrel{(2)}{=} b_1; \dots; b_{\ell}; c_1; \dots; c_{n'-\ell}$, where (1) follows from (*), and (2) by substituting $a_1; \dots; a_k = b_1; \dots; b_{\ell}$.
2. $([g, c_{n-k+1}, \dots], \mathcal{A}_i) \in P$ (with (*)) implies $([g', c_{n'-\ell+1}, \dots], \mathcal{A}_i) \in P \subseteq u_{\S}(P)$. This yields $([b_1, \dots, b_{\ell}, \bar{c}], \mathcal{A}) \in s(u_{\S}(P))$, as desired.

838 ■ ($n < k$, i.e., $h_i; g$ “points to somewhere” in $a_1 \dots a_k$):



839

840 We do not generally have an exact matching b_j to point our g' to. Instead, we choose
 841 $g' = g; a_{n+1}; \dots; a_k$ and $n' = \ell$ as the length of the prefix that is factored. It holds
 842 that:

- 843 1. $h_i; g' = h_i; g; a_{n+1}; \dots; a_k = a_1; \dots; a_n; a_{n+1}; \dots; a_k = b_1; \dots; b_\ell = b_1; \dots; b_{n'}$
- 844 2. $([g, a_{n+1}, \dots, a_k, \bar{c}], \mathcal{A}_i) \in P$ (with $(*)$) implies (by recomposing the first $k - n + 1$ ar-
 845 rows) $([g; a_{n+1}; \dots; a_k, \bar{c}], \mathcal{A}_i) = ([g', \bar{c}], \mathcal{A}_i) \in u_{\S}(P)$. This yields $([b_1, \dots, b_\ell, \bar{c}], \mathcal{A}) \in$
 846 $s(u_{\S}(P))$.

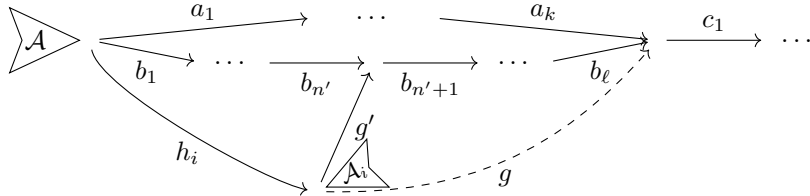
847 ■ (\mathcal{A} is universal (\forall)):

848 Since $([a_1, \dots, a_k, \bar{c}], \mathcal{A}) \in s(P)$ $(*)$, all children $\forall h_i. \mathcal{A}_i$ of \mathcal{A} , all g and n satisfy the
 849 conditions of s . We need to show the same for all children $\forall h_i. \mathcal{A}_i$, all g' and all n' (where
 850 n' is again the length of the prefix being factored) to establish $([b_1, \dots, b_\ell, \bar{c}], \mathcal{A}) \in s(u_{\S}(P))$.
 851 We consider the following subcases, depending on the value of n' :

852 ■ ($n' \geq \ell$ and $b_1; \dots; b_\ell; c_1; \dots; c_{n'-\ell} = h_i; g'$):

853 Since $b_1; \dots; b_\ell = a_1; \dots; a_k$, we have that $a_1; \dots; a_k; c_1; \dots; c_{n'-\ell} = h_i; g'$, which (with
 854 $(*)$) implies that $([g', c_{n-k+1}, \dots], \mathcal{A}_i) \in P \subseteq u_{\S}(P)$.

855 ■ ($n' < \ell$, and $b_1; \dots; b_{n'} = h_i; g'$):



856

857 This yields that $b_1; \dots; b_\ell = h_i; g'; b_{n'+1}; \dots; b_\ell$, and with $(*)$ we can infer that
 858 $([g'; b_{n'+1}; \dots; b_\ell, \bar{c}], \mathcal{A}_i) \in P$. Hence $([g', b_{n'+1}, \dots, b_\ell, \bar{c}], \mathcal{A}_i) \in u_{\S}(P)$.

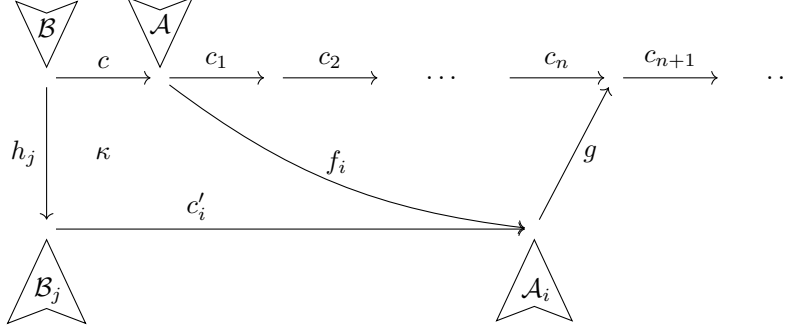
859 Combining both cases, we obtain $([b_1, \dots, b_\ell, \bar{c}], \mathcal{A}) \in s(u_{\S}(P))$, as desired. ◀

860 ► **Lemma 44 (Up-to shift).** Let $P \subseteq \text{Seq} \times \text{Cond}$. Then, $u_{\downarrow}(P)$ is its closure under shift, that
 861 is, $u_{\downarrow}(P) = \{([c; c_1], c_2, \dots], \mathcal{B}) \mid ([c_1, c_2, \dots], \mathcal{B}_{\downarrow c}) \in P\}$. The function u_{\downarrow} is s -compatible.

862 **Proof.** We show that $([c; c_1], c_2, \dots], \mathcal{B}) \in u_{\downarrow}(s(P))$ implies $([c; c_1], c_2, \dots], \mathcal{B}) \in s(u_{\downarrow}(P))$.
 863 Let the children of \mathcal{B} be $\mathcal{Q}h_j. \mathcal{B}_j$ for $j \in J$ and $\mathcal{Q} \in \{\forall, \exists\}$.

864 Since $([c; c_1], c_2, \dots], \mathcal{B}) \in u_{\downarrow}(s(P))$, we know there exists some arrow c such that
 865 $([c_1, c_2, \dots], \mathcal{B}_{\downarrow c}) \in s(P)$. We will call that condition $\mathcal{B}_{\downarrow c} = \mathcal{A}$ and its children $\mathcal{Q}f_i. \mathcal{A}_i$ for
 866 $i \in I$. As we know it results from a shift ($\mathcal{A} = \mathcal{B}_{\downarrow c}$), hence each child $\mathcal{Q}f_i. \mathcal{A}_i$ of \mathcal{A} is derived
 867 from a child $\mathcal{Q}h_j. \mathcal{B}_j$ of \mathcal{B} , that is $\mathcal{A}_i = (\mathcal{B}_j)_{\downarrow c'_i}$, where $(c'_i, f_i) \in \kappa(h_j, c)$ is a representative
 868 square. Then we make the following case distinction:

869 ■ (\mathcal{A} is existential ($\mathcal{Q} = \exists$)):



871 Since $([c_1, c_2, \dots], \mathcal{A}) \in s(P)$, by definition of s there exists some child $\exists f_i. \mathcal{A}_i$ of \mathcal{A} and
 872 g, n such that $c_1; \dots; c_n = f_i; g$ and $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$. (*₁)

873 Because of $([c_1, c_2, \dots], \mathcal{A}) = ([c_1, c_2, \dots], \mathcal{B}_{\downarrow c})$, we know that child $\exists f_i. \mathcal{A}_i$ has been created
 874 from some specific child $\exists h_j. \mathcal{B}_j$ of \mathcal{B} (*₃) such that $\mathcal{A}_i = \mathcal{B}_{j \downarrow c'_i}$ (*₂) and $(c'_i, f_i) \in \kappa(h_j, c)$
 875 (which in particular means that $h_j; c'_i = c; f_i$). (*₄)

876 Using that, we can show $([c; c_1], c_2, \dots, \mathcal{B}) \in s(u_{\downarrow}(P))$:

- 877 ■ *there exists a child $\exists h_j. \mathcal{B}_j$ of \mathcal{B} , and g', n' :* Implied by (*₃). We simply choose the
 878 same child $\exists h_j. \mathcal{B}_j$ and set $g' = c'_i; g$. Also, we simply use $n' = n$.
- 879 ■ *such that $(c; c_1); c_2; \dots; c_{n'} = h_j; g'$:*

$$\begin{array}{ll}
 \text{(by (*}_1\text{))} & c_1; \dots; c_n = f_i; g \\
 \text{(} n = n' \text{)} & c_1; \dots; c_{n'} = f_i; g \\
 \implies & c; c_1; \dots; c_{n'} = c; f_i; g \\
 \text{(by (*}_4\text{))} & = h_j; c'_i; g \\
 \text{(choice of } g') & = h_j; g'
 \end{array}$$

885 ■ *and $([g', c_{n'+1}, \dots], \mathcal{B}_j) \in u_{\downarrow}(P)$:*

$$\begin{array}{ll}
 \text{(*}_1\text{)} & ([g, c_{n+1}, \dots], \mathcal{A}_i) \in P \\
 \text{(*}_2\text{)} \implies & ([g, c_{n+1}, \dots], (\mathcal{B}_j)_{\downarrow c'_i}) \in P \\
 \text{(Def. } u_{\downarrow}\text{)} \implies & ([\underbrace{c'_i; g}_{g'}, \underbrace{c_{n+1}}_{=c_{n'+1}}, \dots], \mathcal{B}_j) \in u_{\downarrow}(P)
 \end{array}$$

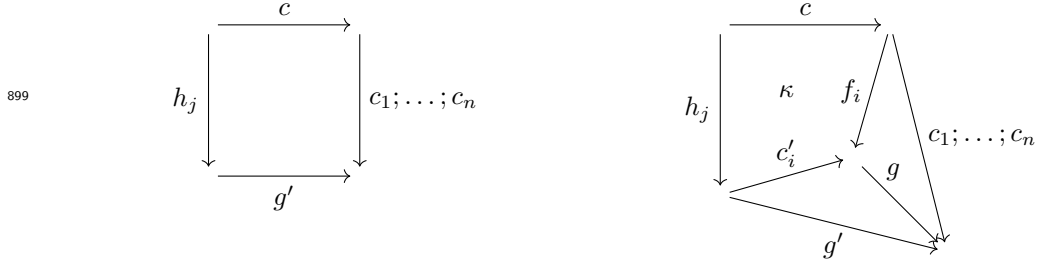
889 ■ (\mathcal{A} is universal ($\mathcal{Q} = \forall$)):

890 Since $([c_1, c_2, \dots], \mathcal{A}) \in s(P)$, by definition of s for each child $\forall f_i. \mathcal{A}_i$ of \mathcal{A} and g, n such
 891 that $c_1; \dots; c_n = f_i; g$, we have $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$.

892 By $([c_1, c_2, \dots], \mathcal{A}) = ([c_1, c_2, \dots], \mathcal{B}_{\downarrow c})$, we know that each such $\forall f_i. \mathcal{A}_i$ has been created
 893 from some specific child $\forall h_j. \mathcal{B}_j$ of \mathcal{B} such that $\mathcal{A}_i = \mathcal{B}_{j \downarrow c'_i}$ and $(c'_i, f_i) \in \kappa(h_j, c)$.

894 We will now show $([c; c_1], c_2, \dots, \mathcal{B}) \in s(u_{\downarrow}(P))$ as follows. Let some child $\forall h_j. \mathcal{B}_j$ of \mathcal{B} ,
 895 g' and n' be given, and assume that $(c; c_1); c_2; \dots; c_{n'} = h_j; g'$. We need to show that
 896 $([g', c_{n'+1}, \dots], \mathcal{B}_j) \in u_{\downarrow}(P)$.

897 Consider the square shown below on the left. According to the assumption from above
 898 and by associativity of composition, it is a commuting square.



Furthermore, it can be reduced to a representative square as shown above on the right. Since the construction of the condition \mathcal{A} involves the same set of squares $\kappa(h_j, c)$, we can associate to our given child $\forall h_j. \mathcal{B}_j$ of \mathcal{B} and g' a particular child $\forall f_i. \mathcal{A}_i$ of \mathcal{A} and g as shown in the diagram above.

We now evaluate the definition of $([c_1, c_2, \dots], \mathcal{A}) \in s(P)$ for that $\forall f_i. \mathcal{A}_i$, g , and $n' = n$. According to that, we have: if $c_1; \dots; c_n = f_i; g$, then $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$. As can be seen in the representative square diagram above, the premise is indeed true, hence $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$ holds as well.

By construction of \mathcal{A} , we have $\mathcal{A}_i = \mathcal{B}_{j \downarrow c'_i}$ and therefore $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$ implies $([g', c_{n+1}, \dots], \mathcal{B}_j) = ([c'_i; g, c_{n+1}, \dots], \mathcal{B}_j) \in u_\downarrow(P)$, which was to be shown. \blacktriangleleft

► **Lemma 45** (Up-to iso). *Let $P \subseteq \text{Seq} \times \text{Cond}$. Let $u_\cong(P)$ be its closure under isomorphic conditions, i.e., $u_\cong(P) = \{([h; c_1], c_2, \dots], \mathcal{B}) \mid ([c_1, c_2, \dots], \mathcal{A}) \in P, \mathcal{A} \cong \mathcal{B} \text{ with iso } h: \text{RO}(\mathcal{B}) \rightarrow \text{RO}(\mathcal{A})\}$. The function u_\cong is s-compatible.*

Proof. We show that $([h; c_1], c_2, \dots], \mathcal{B}) \in u_\cong(s(P))$ implies $([h; c_1], c_2, \dots], \mathcal{B}) \in s(u_\cong(P))$. Let the children of \mathcal{B} be $\mathcal{Q}g_j. \mathcal{B}_j$ for $j \in J$ and $\mathcal{Q} \in \{\forall, \exists\}$.

Since $([h; c_1], c_2, \dots], \mathcal{B}) \in u_\cong(s(P))$, we know there exists some condition \mathcal{A} , having children $\mathcal{Q}f_i. \mathcal{A}_i$ for $i \in I$, such that $\mathcal{A} \cong \mathcal{B}$ wrt. h and $([c_1, c_2, \dots], \mathcal{A}) \in s(P)$.

As $\mathcal{A} \cong \mathcal{B}$ wrt. h , for each $i \in I$ there is $j \in J$ such that $\mathcal{A}_i \cong \mathcal{B}_j$ wrt. an iso $h_{j,i}$ and $h; f_i = g_j; h_{j,i}$; and vice versa (for each $j \in J \dots$).

Now we can show $([h; c_1], c_2, \dots], \mathcal{B}) \in s(u_\cong(P))$ via the following case distinction:

■ **(\mathcal{A} is existential ($\mathcal{Q} = \exists$)):**

Since $([c_1, c_2, \dots], \mathcal{A}) \in s(P)$, by definition of s there exists some child $\exists f_i. \mathcal{A}_i$ of \mathcal{A} and g, n such that $c_1; \dots; c_n = f_i; g$ and $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$.

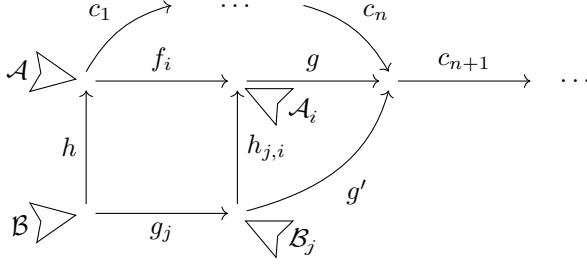
Then we can show that $([h; c_1], c_2, \dots], \mathcal{B}) \in s(u_\cong(P))$:

- *there exists a child $\exists g_j. \mathcal{B}_j$ of \mathcal{B} , and g', n' :* As $\mathcal{A} \cong \mathcal{B}$ wrt. h , there is some child $\exists g_j. \mathcal{B}_j$ of \mathcal{B} such that $\mathcal{B}_j \cong \mathcal{A}_i$ for some iso $h_{j,i}: \text{RO}(\mathcal{B}_j) \rightarrow \text{RO}(\mathcal{A}_i)$ such that $h; f_i = g_j; h_{j,i}$. Let $g' := h_{j,i}; g$ and $n' := n$.
- *such that $h; c_1; \dots; c_{n'} = g_j; g'$:*

$$h; c_1; \dots; c_n \stackrel{(1)}{=} h; f_i; g \stackrel{(2)}{=} g_j; h_{j,i}; g = g_j; g'$$

(1) from $c_1; \dots; c_n = f_i; g$, (2) from $h; f_i = g_j; h_{j,i}$.

- *and $([g', c_{n+1}, \dots], \mathcal{B}_j) \in u_\cong(P)$:* As $\mathcal{A}_i \cong \mathcal{B}_j$ with iso $h_{j,i}$, hence $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$ implies $([h_{j,i}; g], c_{n+1}, \dots], \mathcal{B}_j) \in u_\cong(P)$.



932

933 **■ (\mathcal{A} is universal ($\mathcal{Q} = \forall$)):**

934 We need to show for all children $\forall g_j.\mathcal{B}_j$ of \mathcal{B} and g', n' that: if $h; c_1; \dots; c_{n'} = g_j; g'$ then

935 $([g', c_{n+1}, \dots], \mathcal{B}_j) \in u_{\cong}(P)$.

936 Let such a child, g', n' be given and let $h; c_1; \dots; c_{n'} = g_j; g'$.

937 From $\mathcal{A} \cong \mathcal{B}$ wrt. h we know there is a corresponding child $\forall f_i.\mathcal{A}_i$ of \mathcal{A} , and isos

938 $h: \text{RO}(\mathcal{B}) \rightarrow \text{RO}(\mathcal{A})$ and $h_{j,i}: \text{RO}(\mathcal{B}_j) \rightarrow \text{RO}(\mathcal{A}_i)$ such that $h; f_i = g_j; h_{j,i}$.

939 Now since $h; c_1; \dots; c_{n'} = g_j; g'$, then also $c_1; \dots; c_n = h^{-1}; h; c_1; \dots; c_{n'} = h^{-1}; g_j; g' =$

940 $f_i; h_{j,i}^{-1}; g'$.

941 As $([c_1, c_2, \dots], \mathcal{A}) \in s(P)$, we know that for all children and all g, n (in particular, for

942 $g = h_{j,i}^{-1}; g'$ and $n = n'$) we have: if $c_1; \dots; c_n = f_i; g$ then $([g, c_{n+1}, \dots], \mathcal{A}_i) \in P$.

943 Finally, from $([h_{j,i}^{-1}; g', c_{n+1}, \dots], \mathcal{A}_i) \in P$, using $h_{j,i}$, we obtain $([g', c_{n'+1}, \dots], \mathcal{B}_j) =$

944 $([h_{j,i}; h_{j,i}^{-1}; g', c_{n+1}, \dots], \mathcal{B}_j) \in u_{\cong}(P)$, which implies $([h; c_1, c_2, \dots], \mathcal{B}) \in s(u_{\cong}(P))$.

945 \blacktriangleleft

946 **► Theorem 15 (Up-to techniques).** Let $P \subseteq \text{Seq} \times \text{Cond}$, i.e. tuples of potential model and

947 condition. Then the following four up-to functions are s -compatible:

948 **■** We inductively define a relation \mathcal{U}_{\wedge} containing a pair of conditions $(\mathcal{A}, \mathcal{T})$ iff \mathcal{T} is the

949 same as \mathcal{A} but with some conjunctions removed. That is, \mathcal{U}_{\wedge} contains

$$\bigwedge_{i \in I} \forall f_i.\mathcal{A}_i \quad \mathcal{U}_{\wedge} \quad \bigwedge_{j \in J} \forall f_j.\mathcal{T}_j \quad \text{whenever } (\mathcal{A}_j, \mathcal{T}_j) \in \mathcal{U}_{\wedge} \text{ for all } j \in J$$

$$\bigvee_{i \in I} \exists f_i.\mathcal{A}_i \quad \mathcal{U}_{\wedge} \quad \bigvee_{i \in I} \exists f_i.\mathcal{T}_i \quad \text{whenever } (\mathcal{A}_i, \mathcal{T}_i) \in \mathcal{U}_{\wedge} \text{ for all } i \in I$$

951 Then define up-to conjunction removal: $u_{\wedge}(P) = \{(\bar{c}, \mathcal{T}) \mid (\bar{c}, \mathcal{A}) \in P, (\mathcal{A}, \mathcal{T}) \in \mathcal{U}_{\wedge}\}$

952 **■ Recomposition:** $u_{\S}(P) = \{([b_1, \dots, b_{\ell}, \bar{c}], \mathcal{A}) \mid ([a_1, \dots, a_k, \bar{c}], \mathcal{A}) \in P, a_1; \dots; a_k = b_1; \dots; b_{\ell}\}$

953 **■ Shift:** $u_{\downarrow}(P) = \{([(c; c_1), c_2, \dots], \mathcal{B}) \mid ([c_1, c_2, \dots], \mathcal{B}_{\downarrow c}) \in P\}$

954 **■ Isomorphic condition:** $u_{\cong}(P) = \{([(h; c_1), c_2, \dots], \mathcal{B}) \mid ([c_1, c_2, \dots], \mathcal{A}) \in P, \mathcal{A} \cong \mathcal{B} \text{ with}$

955 $\text{iso } h: \text{RO}(\mathcal{B}) \rightarrow \text{RO}(\mathcal{A})\}$

956 **Proof.** See proofs of Lemmas 42–45 (in the appendix). \blacktriangleleft

957 **► Theorem 17 (Fair branches are models).** Let \mathcal{A}_0 be an alternating condition. Let a fixed

958 tableau constructed by the rules of Definition 11 be given. Let $\mathcal{A}_0 \xrightarrow{b_1} \mathcal{A}_1 \xrightarrow{b_2} \mathcal{A}_2 \xrightarrow{b_3} \dots$ be a

959 branch of the tableau that is either not extendable and ends with a universal quantification

960 (i.e., it is open), or is infinite and fair. For such a branch, we define $P = \{(\bar{b}, \mathcal{A}_i) \mid i \in$

961 $\mathbb{N}_0, \bar{b} = [b_{i+1}, b_{i+2}, \dots]\} \subseteq \text{Seq} \times \text{Cond}$, i.e., the relation P pairs suffixes of the branch with

962 the corresponding conditions. Finally, let u be the combination of up-to techniques defined

963 in Lemma 16. Then, $P \subseteq s(u(P))$, which implies that $P \subseteq \models$. In other words, every such

964 branch in a tableau of Definition 11 corresponds to a model of \mathcal{A}_0 .

965 **Proof.** Let $([c_1, c_2, \dots], \mathcal{C}_0) \in P$, which corresponds to a (suffix of the chosen) branch

966 $\mathcal{A} = \mathcal{C}_0 \xrightarrow{c_1} \mathcal{C}_1 \xrightarrow{c_2} \mathcal{C}_2 \xrightarrow{c_3} \dots$ be of the tableau. We show that $([c_1, c_2, c_3, \dots], \mathcal{A}) \in s(u(P))$.

967 ■ **(\mathcal{A} is existential):**

968 Let $\mathcal{A} = \bigvee_{i \in I} \exists f_i. \mathcal{A}_i$. Hence the first step of our branch (c_1) corresponds to one of the
 969 transitions $\bigvee_{i \in I} \exists f_i. \mathcal{A}_i \xrightarrow{f_\ell} \mathcal{A}_p = \mathcal{C}_1$, where $c_1 = f_\ell$, for some $\ell \in I$. This implies that
 970 $([c_2, \dots], \mathcal{A}_\ell) \in P$.

971 For $([c_1, c_2, \dots], \bigvee_{i \in I} \exists f_i. \mathcal{A}_i) \in s(u(P))$ to hold, by definition of s there must exist some
 972 $i \in I$ (i.e., a child $\exists f_i. \mathcal{A}_i$), an arrow g and a number n such that $c_1; \dots; c_n = f_i; g$
 973 and $([g, c_{n+1}, \dots], \mathcal{A}_i) \in u(P)$. This definition can be satisfied with $n = 1$, $i = \ell$ and
 974 $g = \text{id}$: indeed $c_1 = f_\ell = f_\ell; \text{id}$ and $([c_2, \dots], \mathcal{A}_\ell) \in P$ implies (up-to recomposition)
 975 $([g, c_{1+1}, \dots], \mathcal{A}_\ell) = ([\text{id}, c_2, \dots], \mathcal{A}_\ell) \in u(P)$.

976 Note that the first step (c_1) is guaranteed to exist since we assume that the branch ends
 977 with a universal quantification (if it ends at all).

978 ■ **(\mathcal{A} is universal and has no isomorphisms):**

979 If \mathcal{A} is an empty universal (i.e., true), $(\bar{c}, \mathcal{A}) \in s(u(P))$ trivially holds, since the definition
 980 of s contains a universal quantification over the set of child conditions, and empty
 981 universals are always true.

982 In the following, let $\mathcal{A} = \bigwedge_i \forall f_i. \mathcal{A}_i$. If no f_i is an isomorphism, the tableau does not have
 983 any child nodes for \mathcal{A} , so the sequence $[c_1, \dots]$ is empty, and thus its composition equals
 984 id . Then, $([], \mathcal{A}) \in s(u(P))$ holds because there is no arrow g that makes $\text{id} = f_i; g$ true
 985 (id is an isomorphism and hence can only split in two other isomorphisms, however, f_i is
 986 not an isomorphism). Thus the implication is trivially true.

987 ■ **(\mathcal{A} is universal and has at least one isomorphism):**

988 We now assume that at least one f_i is an isomorphism.

989 For $([c_1, c_2, \dots], \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i) \in s(u(P))$ to hold, by definition of s , for all $\forall f_i. \mathcal{A}_i$, all
 990 arrows g and all $n \in \mathbb{N}_0$, we need to show: if $c_1; \dots; c_n = f_i; g$, then $([g, c_{n+1}, \dots], \mathcal{A}_i) \in$
 991 $u(P)$.

992 Hence let $\forall d_0. \mathcal{D}_0$ (where $d_0 = f_j$, $\mathcal{D}_0 = \mathcal{A}_j$ for some index j) be such a child and assume
 993 some n, g_0 such that $c_1; \dots; c_n = d_0; g_0$. We need to show that $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$.
 994 Since at least one child is an isomorphism, in the next step on the branch, some $\forall f_\ell. \mathcal{A}_\ell$
 995 (f_ℓ isomorphism) is pulled forward, so the next condition on the path has the shape
 996 $\bigvee_j \exists h_j. (\mathcal{B}_j \wedge \dots)$, and it is (up-to conjunction removal) the same as the child $\mathcal{A}_\ell =$
 997 $\bigvee_j \exists h_j. \mathcal{B}_j$ that was pulled forward. If the child to be satisfied is the one that was
 998 pulled forward ($\forall d_0. \mathcal{D}_0 = \forall f_\ell. \mathcal{A}_\ell$), we would thus immediately obtain the desired result
 999 $([\dots], \mathcal{D}_0) \in u(P)$.

1000 However, the $\forall f_\ell. \mathcal{A}_\ell$ that is pulled forward at the current step might be a different
 1001 child than $\forall d_0. \mathcal{D}_0$. While the fairness property would guarantee that every isomorphism
 1002 is eventually pulled forward at some point in the future, d_0 might not even be an
 1003 isomorphism.

1004 So assume that instead of d_0 , for the initial steps of the path, other isomorphisms are
 1005 pulled forward instead. Since the condition is alternating and the tableau rules preserve
 1006 this property, we refer to $c_1, c_2, c_3, c_4, \dots$ as $u_1, e_1, u_2, e_2, \dots$, with u_i, e_i corresponding
 1007 to the labels of steps from a universal or existential condition, respectively (i.e., u_1 was
 1008 pulled forward first). That is $u_i = c_{2i-1}$, $e_i = c_{2i}$.

1009 For each child not yet pulled forward (such as $\forall d_0. \mathcal{D}_0$), the condition \mathcal{C}_2 at the next
 1010 universal step contains successors (such as the children of $(\forall d_0. \mathcal{D}_0)_{\downarrow u_1; e_1}$), which are
 1011 possibly “closer” to an isomorphism than d_0 was.

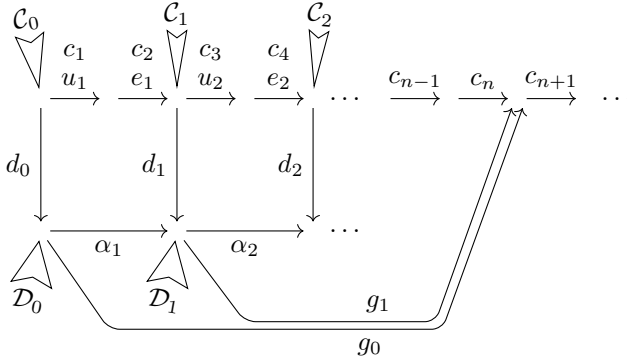
1012 We now claim that there exists a finite sequence $\forall d_0. \mathcal{D}_0, \forall d_1. \mathcal{D}_1, \dots, \forall d_p. \mathcal{D}_p, \dots, \forall d_q. \mathcal{D}_q$
 1013 of children of the universal conditions $\mathcal{C}_0, \mathcal{C}_2, \dots$ on the branch (i.e., every second one),
 1014 with each element of the sequence being a (direct) successor of the previous element, such

1015 that:

- 1016 1. After p steps, $0 \leq 2p \leq n$, d_p is an isomorphism,
- 1017 2. for $k > p$, d_k is an isomorphism as well,
- 1018 3. after q steps, $p \leq q < \infty$, d_q is pulled forward, resulting in $([e_{q+1}, \dots], \mathcal{D}_q) \in u(P)$.

1019 Afterwards, we can transform that to $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$ (as required by the
 1020 satisfaction function s) by applying up-to techniques, thereby showing that the path
 1021 actually describes a model.

1022 Note that $p = 0$ or $p = q$ are possible as well. The initial steps are depicted in the
 1023 following commuting diagram, with the chain u_1, e_1, \dots, n , d_0, \mathcal{D}_0 and g_0 given. The
 1024 remaining conditions and arrows will now be constructed.

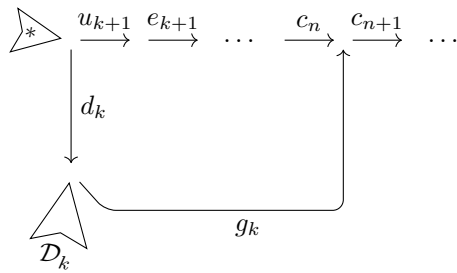


1025

1026 The proof objective is to construct the sequence of all further $\forall d_{k+1}. \mathcal{D}_{k+1}$ and associated
 1027 α_{k+1}, g_{k+1} .

- 1028 1. We construct the aforementioned sequence by repeatedly choosing a successor of $\forall d_k. \mathcal{D}_k$
 1029 as the next element ($\forall d_{k+1}. \mathcal{D}_{k+1}$) of the sequence. We then show that for some p with
 1030 $2p \leq n$, d_p must be an isomorphism.

1031 Hence, consider the following situation:



1032

1033 The condition labeled $*$ on the top left corresponds to \mathcal{C}_{2k} . It is of the form
 1034 $\left(\bigwedge_{m \neq j} \forall f_m. \mathcal{A}_m \right) \wedge \forall d_k. \mathcal{D}_k$, i.e., one of its children is the current element $\forall d_k. \mathcal{D}_k =$
 1035 $\forall f_j. \mathcal{A}_j$ of the sequence, and further children $\forall f_m. \mathcal{A}_m$.

1036 If d_k is an isomorphism, we are done, set $p = k$ and we do not need to choose a next
 1037 element. We will show that this point is reached eventually and that $2p \leq n$ later in
 1038 the proof.

1039 So assume that d_k is not an isomorphism. So some other arrow f_ℓ was pulled forward
 1040 in this step instead.

XX:30 Coinductive Techniques for Checking Satisfiability of Generalized Nested Conditions

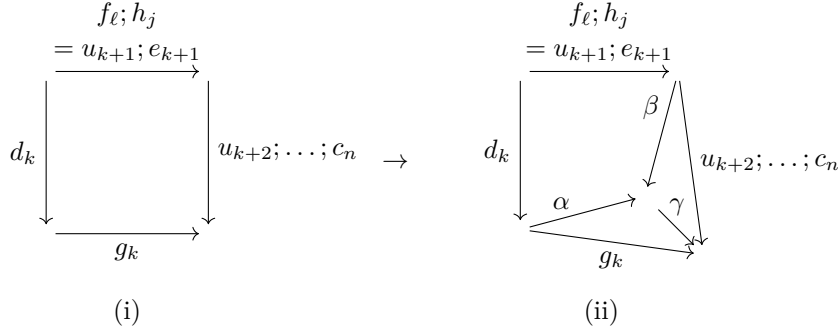
1041 = Assume u_{k+1} is not c_n and there is at least one more arrow e_{k+1} before c_n is reached.
 1042 Let $\forall f_\ell.\mathcal{A}_\ell$ be the child that was chosen in the tableau to be pulled forward now
 1043 (i.e., $f_\ell = u_{k+1}$ is an isomorphism). Then, the condition \mathcal{C}_{2k} at the current node
 1044 (marked * in the diagram above) is $\left(\bigwedge_{m \neq \ell} \forall f_m.\mathcal{A}_m\right) \wedge \forall f_\ell.\mathcal{A}_\ell$, with $\forall d_k.\mathcal{D}_k$ being
 1045 among the $\forall f_m.\mathcal{A}_m$, $m \neq \ell$. Assume the condition that was pulled forward has the
 1046 structure $\mathcal{A}_\ell = \bigvee_j \exists h_j.\mathcal{B}_j$, which results in a step:

$$1047 \quad \left(\bigwedge_{m \neq \ell} \forall f_m.\mathcal{A}_m\right) \wedge \forall f_\ell.\mathcal{A}_\ell \xrightarrow[=u_{k+1}]{f_\ell} \underbrace{\bigvee_j \exists h_j.\mathcal{B}_j}_{\mathcal{A}_\ell} \wedge \underbrace{\left(\bigwedge_{m \neq \ell} \forall f_m.\mathcal{A}_m\right)_{\downarrow f_\ell; h_j}}_{\text{contains successors of } \forall d_k.\mathcal{D}_k}$$

1048 Since the next condition is an existential, it has outgoing transitions for each h_j ,
 1049 with one of them corresponding to the e_{k+1} that is next on the path:

$$1050 \quad \bigvee_j \exists h_j.\left(\mathcal{B}_j \wedge \left(\bigwedge_{m \neq \ell} \forall f_m.\mathcal{A}_m\right)_{\downarrow f_\ell; h_j}\right) \xrightarrow[=e_{k+1}]{h_j} \mathcal{B}_j \wedge \underbrace{\left(\bigwedge_{m \neq \ell} \forall f_m.\mathcal{A}_m\right)_{\downarrow f_\ell; h_j}}_{\text{contains successors of } \forall d_k.\mathcal{D}_k}$$

1051 We now reinterpret the diagram from before as a square, shown below in (i). The
 1052 right side of square (i) is the composition of all remaining arrows until c_n . In case
 1053 e_{k+1} is actually c_n and there are no further arrows left, the right side is simply id.

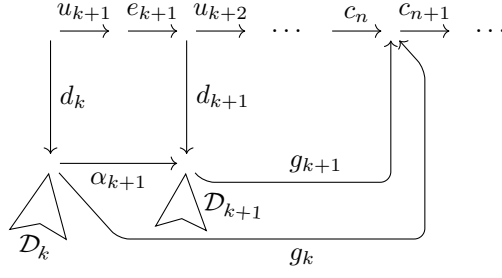


1054 Being a commuting square, (i) can be reduced to some representative square
 1055 $(\alpha, \beta) \in \kappa(d_k, (u_{k+1}; e_{k+1})) = \kappa(d_k, (f_\ell; h_j))$ (shown above in square (ii) on the
 1056 right). Depending on κ , there might be several pairs of α, β that make (ii) commute,
 1057 in which case an arbitrary one can be chosen.

1058 The shift that happens in the current condition (in particular that of $\forall d_k.\mathcal{D}_k$) refers
 1059 to the same set of squares:
 1060

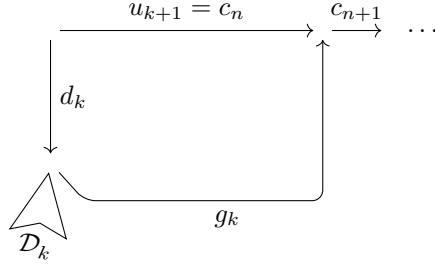
$$1061 \quad (\forall d_k.\mathcal{D}_k)_{\downarrow f_\ell; h_j} = \bigwedge_{(\alpha, \beta) \in \kappa(d_k, (f_\ell; h_j))} \forall \beta.(\mathcal{D}_k)_{\downarrow \alpha}$$

1062 This means that one of the successors of $\forall d_k.\mathcal{D}_k$ actually uses the representative
 1063 square of (ii) above. Now let α, β be the pair of arrows that close that square
 1064 and γ be the mediating morphism. Then rename $\alpha_{k+1} = \alpha$, $d_{k+1} = \beta$, $g_{k+1} = \gamma$,
 1065 $\mathcal{D}_{k+1} = (\mathcal{D}_k)_{\downarrow \alpha_{k+1}}$ (note that $(\mathcal{D}_k)_{\downarrow \alpha_{k+1}}$ is one of the successors of \mathcal{D}_k) and extend
 1066 our original diagram:



Now start over from the beginning, with d_{k+1}, g_{k+1} instead of d_k, g_k . (Since the chain u_{k+2}, \dots, c_n is now two elements shorter than before, this process will not continue endlessly.)

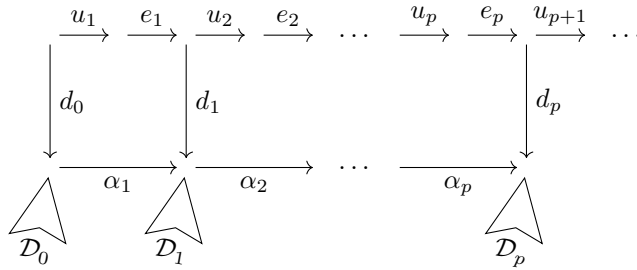
- Consider the case that $c_n = u_{k+1}$:



Labels of transitions from a universal condition (such as u_{k+1}) are always isomorphisms. Hence $u_{k+1} = d_k; g_k$ can only split into isomorphisms, so d_k must be an isomorphism as well.

- Consider the case that $c_n = e_{k+1}$ or $n = 0$: In both cases the remaining sequence c_{n+1}, \dots, c_n is empty, representing the identity id. Hence $d_{k+1}; g_{k+1} = \text{id}$, which implies that d_{k+1} is an iso.
(This subcase and the previous one ensure that an isomorphism is obtained after finitely many – at most p with $2p \leq n$ – steps.)

- Summarizing the situation so far, the original morphism d_0 has been “turned into an iso” d_q , more precisely there is a successor $d_p \cdot \mathcal{D}_p$ of $d_0 \cdot \mathcal{D}_0$ that is an iso:



To be able to pull an indirect successor of d_p forward, it has to be an isomorphism. Generally, for $k \geq p$, not all direct successors of $\forall d_k \cdot \mathcal{D}_k$ might have isomorphisms (depending on the class κ in use). However, shifting isomorphisms always results in at least one isomorphism again (cf. Lemma 7), so at least one successor of $\forall d_k \cdot \mathcal{D}_k$ contains an isomorphism. Hence for $\forall d_{k+1} \cdot \mathcal{D}_{k+1}$ we choose an arbitrary one of these, and thereby ensure that all elements $\forall d_{p+1} \cdot \mathcal{D}_{p+1}, \forall d_{p+2} \cdot \mathcal{D}_{p+2}, \dots$ of the sequence do in fact have isomorphisms.

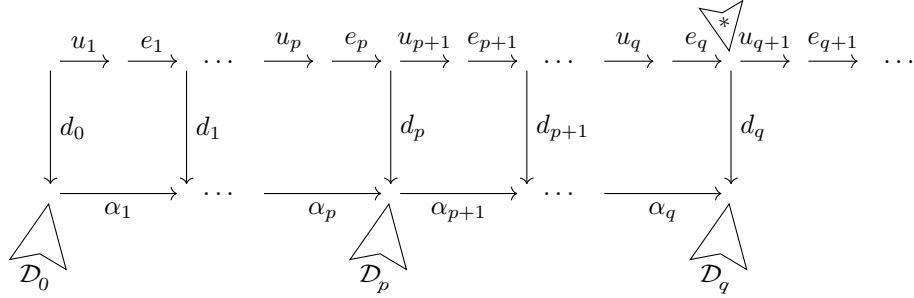
Furthermore by construction we have that $u_{i+1}; e_{i+1}; d_{i+1} = d_k; \alpha_{i+1}$ for all i which means that the row of squares in the diagram above commutes. From the tri-

angles in the representative square diagram we obtain that $g_i = \alpha_{i+1}; g_{i+1}$ and $d_{i+1}; g_{i+1} = u_{i+2}; \dots; c_n$. Whenever c_n is a morphism derived from an existential step ($c_n = e_{k+1}$), we have that $d_{k+1}; g_{k+1} = \text{id}$ (see above). And whenever c_n is a morphism derived from a universal step ($c_n = u_{k+1}$), we know that $d_k; g_k = u_{k+1}$ and $\alpha_{k+1} = g_k; c_{n+1}; d_{k+1}$. The latter is true since $d_k; \alpha_{k+1} = u_{k+1}; e_{k+1}; d_{k+1} = d_k; g_k; e_{k+1}; d_{k+1} = d_k; g_k; c_{n+1}; d_{k+1}$ and d_k is an iso as argued above.

3. Since d_p is an isomorphism, it must eventually be pulled forward. We continue the construction of the morphisms, potentially beyond c_n , and choose the representative squares such that the arrows d_k with $k \geq p$ are also isos (cf. the assumption after Lemma 7).

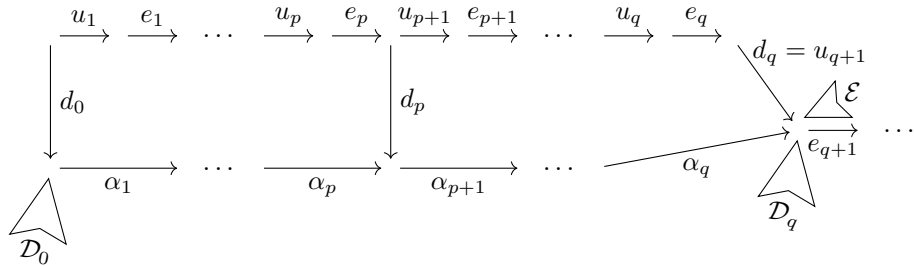
If the path is infinite, the fairness constraint guarantees the existence of an index q where an iso d_q which is a successor of d_p is pulled forward. If the path is finite, the length of the path is an upper bound for the value of q , and some indirect successor of d_p must have been pulled forward by then: the only way that a branch can be unextendable is that there are no isomorphisms that could be pulled forward; however, for $k \geq p$, d_k is an isomorphism (as shown in the previous item), so the path cannot end before one of these d_k has actually been pulled forward.

This pull-forward step eventually happens at condition \mathcal{C}_{2q} , being marked by $*$ in the diagram. Both cases $2q \leq n$ and $2q > n$ are possible. We assume that $\mathcal{C}_{2q} = \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i$.



Hence at step q a successor $\forall d_q. \mathcal{D}_q$ of $\forall d_p. \mathcal{D}_p$ is finally pulled forward, i.e., the next label u_{q+1} equals d_q . Assume that $\forall d_q. \mathcal{D}_q = \forall f_\ell. \mathcal{A}_\ell$. Let $\mathcal{D}_q = \bigvee_j \exists h_j \mathcal{B}_j$, then the next step is:

$$\left(\bigwedge_{m \neq \ell} \forall f_m. \mathcal{A}_m \right) \wedge \forall d_q. \mathcal{D}_q \xrightarrow[=u_{q+1}]{d_q} \overbrace{\bigvee_j \exists h_j. \left(\mathcal{B}_j \wedge \left(\bigwedge_{m \neq q} \forall f_m. \mathcal{A}_m \right)_{\downarrow d_q; h_j} \right)}^{\mathcal{E}}$$



As a result, we have: $([e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{E}) \in P$.

1120 Note that the subsequence u_{q+2}, e_{q+2}, \dots may or may not be empty. (However,
 1121 e_{q+1} exists because the path cannot end at an empty existential.) Furthermore
 1122 $d_q; g_q = u_{q+1}; e_{q+1}; \dots; c_n = d_q; e_{q+1}; \dots; c_n$, which implies that $g_q = e_{q+1}; \dots; c_n$,
 1123 since d_q is an iso.

1124 We now apply our up-to techniques to obtain the originally desired result (given $\forall d_0. \mathcal{D}_0$,
 1125 n and g_0 such that $c_1; \dots; c_n = d_0; g_0$, show that $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$).

1126 Observe that the root objects of \mathcal{D}_q and \mathcal{E} coincide. They are not the same condition,
 1127 however, \mathcal{D}_q is “contained” in \mathcal{E} , in the sense that the path condition \mathcal{E} is a (nested)
 1128 conjunction of \mathcal{D}_q and additional conditions. Hence we first apply up-to conjunction
 1129 removal (Lemma 42) such that our condition is not \mathcal{E} (“ \mathcal{D}_q with conjunctions”), but only
 1130 \mathcal{D}_q :

$$1131 \quad \Rightarrow ([e_{q+1}, u_{q+1}, e_{q+2}, \dots], \bigvee_j \exists h_j. \mathcal{B}_j) = ([e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_q) \in u(P)$$

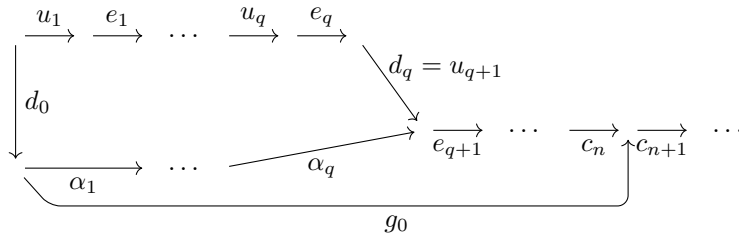
1132 Since $\mathcal{D}_q = (\mathcal{D}_{q-1})_{\downarrow \alpha_q} = \dots = (\mathcal{D}_0)_{\downarrow \alpha_1 \downarrow \dots \downarrow \alpha_q}$, we can now apply a series of up-to shift
 1133 operations (Lemma 44) to obtain a tuple with \mathcal{D}_0 as the condition:

$$\begin{aligned} 1134 & ([e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_0_{\downarrow \alpha_1 \downarrow \dots \downarrow \alpha_q}) \in u(P) \\ 1135 & \Rightarrow ([\alpha_q; e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_0_{\downarrow \alpha_1 \downarrow \dots \downarrow \alpha_{q-1}}) \in u(P) \\ 1136 & \Rightarrow \dots \Rightarrow ([\alpha_1; \dots; \alpha_q; e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_0) \in u(P) \end{aligned}$$

1137 Now we use up-to recomposition (Lemma 43) to rewrite the initial part of the chain,
 1138 depending on where c_n is in the path, relative to d_q :

1139

1140 = $(2q \leq n, \text{ i.e., } c_n \text{ is after } d_q)$:



1141

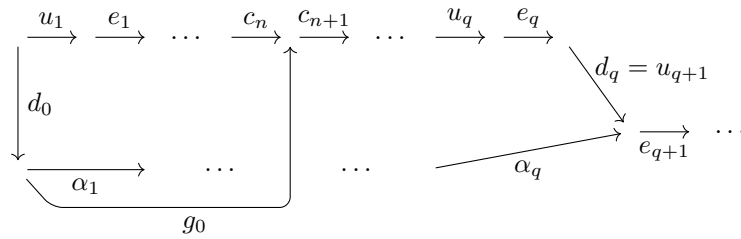
1142 As indicated in the diagram, we have constructed arrows in such a way that

$$1143 \quad g_0 = \alpha_1; g_1 = \dots = \alpha_1; \dots; \alpha_q; g_q = \alpha_1; \dots; \alpha_q; e_{q+1}; u_{q+2}; e_{q+2}; \dots; c_n.$$

1144 Hence

$$\begin{aligned} 1145 & ([\alpha_1; \dots; \alpha_q; e_{q+1}, u_{q+2}, e_{q+2}, \dots, c_n, c_{n+1}, \dots], \mathcal{D}_0) \in u(P) \\ 1146 & \Rightarrow ([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P) \end{aligned}$$

1147 ■ $(2q > n, \text{ i.e., } c_n \text{ is before } d_q):$



1148

As indicated in the diagram, we can show that $g_0; c_{n+1}; \dots; u_{q+1} = \alpha_1; \dots; \alpha_q$. In order to prove this equality, we have to make a case distinction:

* Whenever c_n originates from a universal step, that is $c_n = u_{k+1}$, we know that c_{n+1} stems from an existential step, that is $c_{n+1} = e_{k+1}$. In this case we use the fact that the d_ℓ for $\ell \geq k+1$ are isos:

$$\begin{aligned}
1154 \quad & g_0; c_{n+1}; \dots; u_{q+1} = \alpha_0; \dots; \alpha_k; g_k; c_{n+1}; \dots; u_{q+1} \\
1155 \quad & = \alpha_0; \dots; \alpha_k; \alpha_{k+1}; d_{k+1}^{-1}; c_{n+2}; \dots; u_{q+1} \\
1156 \quad & = \alpha_0; \dots; \alpha_k; \alpha_{k+1}; \alpha_{k+2}; d_{k+2}^{-1}; c_{n+4}; \dots; u_{q+1} = \dots \\
1157 \quad & = \alpha_1; \dots; \alpha_q; d_q^{-1}; u_{q+1} \\
1158 \quad & = \alpha_1; \dots; \alpha_q; d_q^{-1}; d_q = \alpha_1; \dots; \alpha_q
\end{aligned}$$

* Whenever c_n originates from an existential step, that is $c_n = e_{k+1}$, we know that c_{n+1} stems from a universal step, that is $c_n = u_{k+2}$. In this case we can infer from $\alpha_{k+1}; g_{k+1} = g_k$ that $\alpha_{k+1} = \alpha_{k+1}; \text{id} = \alpha_{k+1}; g_{k+1}; d_{k+1} = g_k; d_{k+1}$. This implies:

$$\begin{aligned}
1162 \quad & g_0; c_{n+1}; \dots; u_{q+1} = \alpha_0; \dots; \alpha_k; g_k; c_{n+1}; \dots; u_{q+1} \\
1163 \quad & = \alpha_0; \dots; \alpha_k; \alpha_{k+1}; d_{k+1}^{-1}; c_{n+1}; \dots; u_{q+1} \\
1164 \quad & = \alpha_0; \dots; \alpha_k; \alpha_{k+1}; \alpha_{k+2}; d_{k+2}^{-1}; c_{n+3}; \dots; u_{q+1} = \dots \\
1165 \quad & = \alpha_1; \dots; \alpha_q; d_q^{-1}; u_{q+1} \\
1166 \quad & = \alpha_1; \dots; \alpha_q; d_q^{-1}; d_q = \alpha_1; \dots; \alpha_q
\end{aligned}$$

1167 Hence

$$\begin{aligned} & ([\alpha_1; \dots; \alpha_q; e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_0) \in u(P) \\ \implies & ([g_0, c_{n+1}, \dots, u_{q+1}, e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_0) \in u(P) \end{aligned}$$

1170 In both cases we have achieved the desired result.

1171 ► **Theorem 18** (Soundness). *If there exists a tableau T for a condition \mathcal{A} where all branches*
1172 *are closed, then the condition \mathcal{A} in the root node is unsatisfiable.*

1173 **Proof.** By induction over the depth of T .

1174 **■** Let the depth of T be 1, i.e., T consists of exactly one node (the root \mathcal{A}). Since all
1175 branches (in this case, the only branch) of T are closed and therefore end with an empty
1176 existential, this node itself, which is also the root node, is an empty existential. An empty
1177 existential condition is unsatisfiable. Therefore, the statement holds.

1178 ■ Assume the statement holds for tableaux of depth d . Let T be a tableau with depth $d + 1$.
1179 T has several children as created by one of the tableau rules (\forall or \exists). Each child can be
1180 seen as the root of a subtableau T_i . As all branches of T are closed, so are all branches of
1181 all T_i , and each T_i has a depth of at most d . Therefore, the root \mathcal{A}_i of each subtableau
1182 T_i is unsatisfiable. Now consider the root of T :

- 1183 ■ Root of T is existential ($\mathcal{A} = \bigvee_i \exists f_i. \mathcal{A}_i$): The children of the root of T are, by
 1184 construction, exactly the children of \mathcal{A} . By definition of satisfaction, \mathcal{A} is satisfiable if
 1185 and only if at least one \mathcal{A}_i is satisfiable. However, all \mathcal{A}_i , being the roots of closed
 1186 tableaux T_i of depth d , are known to be unsatisfiable. Therefore, \mathcal{A} is also unsatisfiable.
- 1187 ■ Root of T is universal ($\mathcal{A} = \bigwedge_i \forall f_i. \mathcal{A}_i$): Since T has depth $d + 1$, the root has a
 1188 single child node \mathcal{A}' that was created by pulling forward some isomorphism f_p from
 1189 \mathcal{A} . \mathcal{A}' is the root of a closed subtableau of depth d and therefore unsatisfiable. Then,
 1190 $\exists f_p. \mathcal{A}'$ is also unsatisfiable, which by Lemma 10 is equivalent to \mathcal{A} and therefore also
 1191 unsatisfiable. ◀

1192 ► **Lemma 21.** *Let \mathcal{A} be a condition and T be a fully expanded tableau for \mathcal{A} . Then,*
 1193 *for each branch it holds that it either is finite, or that there always eventually is another*
 1194 *non-isomorphism on the branch.*

1195 **Proof.** We show that every iso on the branch is followed by finitely many further isos and
 1196 eventually by a non-iso, or the branch ends. Equivalently: for each subtableau T' of T ,
 1197 every branch is finite, or begins with finitely many isos followed by a non-iso. We do this by
 1198 showing that it is impossible for a path to begin with infinitely many isos.

1199 Assume we are starting with a universal condition \mathcal{A} (for an existential condition, simply
 1200 proceed with the universal conditions in its children).

1201 \mathcal{A} either has no isos as children, resulting in no pull-forward step and hence a finite
 1202 branch of length zero, which validates the statement. Otherwise, assume some $\forall f_p. \mathcal{A}_p$ was
 1203 pulled forward, resulting in a single existential child $\bigvee_j \exists g_j. \mathcal{C}_j$.

1204 Of this existential condition, every child where g_j is a non-isomorphism immediately
 1205 validates the statement, as, starting from the root, only one iso (f_p) precedes a non-iso g_j .

1206 Consider a child where g_j is an isomorphism, and the next node on the branch is
 1207 hence a universal condition \mathcal{C}_j . Being a result of a pull-forward step, its shape is $\mathcal{C}_j =$
 1208 $\mathcal{B}_j \wedge \bigwedge_{m \neq p} (\forall f_m. \mathcal{A}_m) \downarrow_{f_p; g_j}$, where $\mathcal{A}_p = \bigvee_\ell \exists g_\ell. \mathcal{B}_\ell$.

1209 We will compare the weights (respectively size) of conditions \mathcal{A} and \mathcal{C}_j , by defining a
 1210 weight function w on conditions as follows:

$$1211 \quad w(\bigvee_i \exists f_i. \mathcal{A}_i) = w(\bigwedge_i \forall f_i. \mathcal{A}_i) = 1 + \sum_i w(\mathcal{A}_i)$$

1212 (Note that the base cases are conditions with no children with $w(\text{true}) = w(\text{false}) = 1$
 1213 according to the definition above.)

1214 Assume $\mathcal{A} = \forall f_p. (\bigvee_{j'} \exists g_{j'}. \mathcal{B}_{j'}) \wedge \bigwedge_{m \neq p} \forall f_m. \mathcal{A}_m$. Let $w_p = \sum_{m \neq p} w(\mathcal{A}_m)$ be the
 1215 combined weight of the children of \mathcal{A} that were *not* pulled forward. Also, for the condition
 1216 \mathcal{B}_j which is part of the conjunction \mathcal{C}_j , assume that $\mathcal{B}_j = \bigwedge_k \forall h_k. \mathcal{D}_k$. Then:

$$\begin{aligned}
 1217 \quad w(\mathcal{A}) &= 1 + w(\bigvee_{j'} \exists g_{j'}. \mathcal{B}_{j'}) + w_p \\
 1218 \quad &= 1 + 1 + \left(\sum_{j'} w(\mathcal{B}_{j'}) \right) + w_p && \text{(includes weights of all } \exists \text{ children)} \\
 1219 \quad &\geq 1 + w(\mathcal{B}_j) + w_p && \text{(only the child on the current branch)} \\
 1220 \quad &= 1 + \left(1 + \sum_k w(\mathcal{D}_k) \right) + w_p
 \end{aligned}$$

1221 Now compute the weight of \mathcal{C}_j :

$$\begin{aligned}
 1222 \quad w(\mathcal{C}_j) &= w(\mathcal{B}_j \wedge \bigwedge_{m \neq p} (\forall f_m. \mathcal{A}_m) \downarrow_{f_p; g_j}) \\
 1223 \quad &= w(\bigwedge_k \forall h_k. \mathcal{D}_k \wedge \bigwedge_{m \neq p} (\forall f_m. \mathcal{A}_m) \downarrow_{f_p; g_j})
 \end{aligned}$$

$$\begin{aligned}
 &= 1 + \left(\sum_k w(\mathcal{D}_k) \right) + \sum_{m \neq p} w(\mathcal{A}_{m \downarrow f_p; g_j}) \\
 &\stackrel{(*)}{=} 1 + \left(\sum_k w(\mathcal{D}_k) \right) + \sum_{m \neq p} w(\mathcal{A}_m) \\
 &= 1 + \left(\sum_k w(\mathcal{D}_k) \right) + w_p
 \end{aligned}$$

The equality marked with $(*)$ holds because $f_p; g_j$ is an isomorphism (f_p because only isos can be pulled forward), and for an iso i , $w(\mathcal{C}_{\downarrow i}) = w(\mathcal{C})$. Since we previously assumed that κ has only a single representative square when isos are involved, then by the assumption made at the end of Section 2.4, when shifting over an isomorphism, the resulting condition has the same structure as the original one (we obtain exactly one square per child condition, and hence the new condition has exactly as many children as the original one; and its child conditions are shifted over identities). Since the weight function considers only child count and nesting depth, shifting over isos preserves the weight.

Clearly $w(\mathcal{A}) > w(\mathcal{C}_j)$.

Hence, if a branch starts at a universal condition, and the next arrow on the path is an isomorphism, the weight of the condition is reduced by at least 1. This excludes having an infinite sequence of isomorphisms at the beginning of the branch, since the weight cannot be reduced infinitely many times. Hence every branch eventually ends, or a non-iso occurs on it after at most $2 \cdot w(\mathcal{A})$ leading isomorphisms. \blacktriangleleft

► **Theorem 22 (Model Finding).** *Let \mathcal{A} be a condition, m a finitely decomposable arrow such that $m \models \mathcal{A}$ and let T be a fully expanded tableau for \mathcal{A} .*

Then, there exists an open and unextendable branch with arrows c_1, \dots, c_n in T , having condition \mathcal{R} in the leaf node, where $m = c_1; \dots; c_n; r$ for some r with $r \models \mathcal{R}$. Furthermore the finite prefix is itself a model for \mathcal{A} (i.e., $[c_1, \dots, c_n] \models \mathcal{A}$).

Proof. First, we observe that by the tableau construction rules of Definition 11, if $m \models \mathcal{A}$, then the tableau T for \mathcal{A} contains at least one branch $\mathcal{A} = \mathcal{C}_0 \xrightarrow{c_1} \mathcal{C}_1 \xrightarrow{c_2} \mathcal{C}_2 \xrightarrow{c_3} \dots$ such for each node \mathcal{C}_i on the branch, it holds that $m = c_1; \dots; c_i; r_i$ and $r_i \models \mathcal{C}_i$. (This is because the tableau contains only equivalence transformations and existential unfolding. The existence of at least one such branch is guaranteed for all models, whether they are infinite or finite.)

This can be shown by induction: assume that we have constructed the path up to index i . We distinguish the following cases:

- $\mathcal{C}_i = \bigvee_{j \in J} \exists f_j. \mathcal{A}_j$ is existential: since $r \models \mathcal{C}_i$, there exists an index $j \in J$ such that $r_i = f_j; r_{i+1}$ for some arrow r_{i+1} with $r_{i+1} \models \mathcal{A}_j$. Then choose $c_{i+1} = f_j$, $\mathcal{C}_{i+1} = \mathcal{A}_j$. Clearly $m = c_1; \dots; c_i; r_i = c_1; \dots; c_i; f_j; r_{i+1} = c_1; \dots; c_i; c_{i+1}; r_{i+1}$.
- $\mathcal{C}_i = \bigwedge_{j \in J} \forall f_j. \mathcal{A}_j$ is universal: then there is some (non-deterministically chosen) tableau step reaching \mathcal{C}_{i+1} , where c_{i+1} is an iso and $\mathcal{C}_i \equiv \exists c_{i+1}. \mathcal{C}_{i+1}$ (see Lemma 10). Define $r_{i+1} = c_{i+1}^{-1}; r_i$, which gives us $m = c_1; \dots; c_i; r_i = c_1; \dots; c_i; c_{i+1}; r_{i+1}$.

A result of this observation is that every model (or a prefix thereof) actually occurs in the tableau. We will now show that such a prefix will occur at the end of a finite branch, and hence be produced as a positive output of the algorithm.

Consider a branch that has the property described above. If there are multiple branches that satisfy this property, pick an arbitrary one. The branch could be either finite or infinite:

- Assume the branch is infinite. By Lemma 21, the branch must contain infinitely many non-isomorphisms c_i . However, m is finitely decomposable by assumption, so such a decomposition is impossible.

Thus, any branch with the property given above cannot be infinite.

1268 ■ Assume the branch is finite and ends with condition $\mathcal{R} = \mathcal{C}_\ell$ in the leaf. Then, \mathcal{R} is not
 1269 an existential condition, since then it would be false and hence has no model.
 1270 Hence \mathcal{R} is a universal condition that has no isomorphisms. Then, $\text{id} \models \mathcal{R}$. ◀

1271 **C** Proofs and Additional Material for §4 (Witnesses for infinite 1272 models)

1273 ▶ **Theorem 27** (Witnesses). *Let \mathcal{C}_0 be an alternating, universal condition. Let a fixed tableau*
 1274 *constructed by the rules of Definition 11 be given. Let $\mathcal{C}_0 \xrightarrow{b_1} \mathcal{C}_1 \xrightarrow{b_2} \dots \xrightarrow{b_r} \mathcal{C}_r$ be a fair segment*
 1275 *of a branch of the tableau where $r > 0$, and let some arrow m be given such that $\mathcal{C}_0 \cong (\mathcal{C}_r)_{\downarrow m}$ for*
 1276 *an iso $\iota: \text{RO}(\mathcal{C}_{r\downarrow m}) \rightarrow \text{RO}(\mathcal{C}_0)$. Then, $[b_1, \dots, b_r, m; \iota]^\omega := [b_1, \dots, b_r, m; \iota, b_1, \dots, b_r, m; \iota, \dots]$*
 1277 *is a model for \mathcal{C}_0 .*

1278 **Proof.** Let a fair segment $\mathcal{C}_0 \xrightarrow{u_1} \mathcal{C}'_0 \xrightarrow{e_1} \mathcal{C}_1 \xrightarrow{u_2} \mathcal{C}'_1 \xrightarrow{e_2} \dots \xrightarrow{e_r} \mathcal{C}_r$ be given, where u_i, e_i
 1279 correspond to labels of universal and existential steps, respectively. That is $u_i = b_{2i-1}$,
 1280 $e_i = b_{2i}$.

1281 Also let an arrow m be given such that $\mathcal{C}_0 \cong \mathcal{C}_{r\downarrow m}$ wrt. an iso ι . Note that \mathcal{C}_0 is universal,
 1282 the condition is alternating, and $\mathcal{C}_{r\downarrow m} \cong \mathcal{C}_0$ implies that \mathcal{C}_r is also universal. Hence r must
 1283 be even. We define the relation

$$P = \{([b_1, b_2, \dots, b_r, m; \iota, (b_1, \dots, b_r, m; \iota)^\omega], \mathcal{C}_0), \\ ([b_2, \dots, b_r, m; \iota, (b_1, \dots, b_r, m; \iota)^\omega], \mathcal{C}_1), \\ \vdots \\ ([m; \iota, (b_1, \dots, b_r, m; \iota)^\omega], \mathcal{C}_r)\}$$

1285 and show that $P \subseteq s(u(P))$: Let $([c_1, c_2, c_3, \dots], \mathcal{A}) \in P$ for some condition \mathcal{A} on the segment.
 1286 We show that $([c_1, c_2, c_3, \dots], \mathcal{A}) \in s(u(P))$. The general structure of the proof matches that
 1287 of Theorem 17. Hence we highlight the differences and refer to Theorem 17 for detailed
 1288 explanations of the unchanged parts.

1289 Compared to Theorem 17, arrows on the paths of P can have a step based on $m; \iota$ that
 1290 does not result from an application of a tableau rule, but serves to close the loop from \mathcal{C}_r
 1291 to \mathcal{C}_0 via the shift $\mathcal{C}_{r\downarrow m}$ (and an isomorphism ι). This requires changes to the selection of
 1292 successors of $\forall d_0. \mathcal{D}_0$. Furthermore, the fairness constraint is guaranteed for the repeated
 1293 segment, which slightly changes the reasoning for when a successor of $\forall d_p. \mathcal{D}_p$ is pulled
 1294 forward.

1295 ■ (\mathcal{A} is existential):

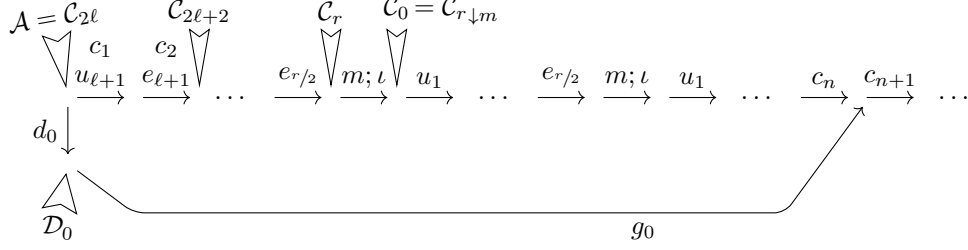
1296 Existential conditions only occur in the middle of the segment, since the step for $m; \iota$
 1297 originates from \mathcal{C}_r which is universal. Hence we can handle this case in exactly the same
 1298 way as in the proof of Theorem 17, which refers to the tuple in P that corresponds to the
 1299 next condition on the path, and satisfies the definition of s using $n = 1$ and $g = \text{id}$.

1300 ■ (\mathcal{A} is universal):

1301 Let $\mathcal{A} = \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i$. For $([c_1, c_2, \dots], \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i) \in s(u(P))$ to hold, by definition of s ,
 1302 for all children $\forall f_i. \mathcal{A}_i$, all arrows g and all $n \in \mathbb{N}_0$, we need to show: if $c_1; \dots; c_n = f_i; g$,
 1303 then $([g, c_{n+1}, \dots], \mathcal{A}_i) \in u(P)$.

1304 Now choose a particular child $\forall d_0. \mathcal{D}_0$ of \mathcal{A} that should be satisfied, some n , and
 1305 some g_0 such that $c_1; \dots; c_n = d_0; g_0$, be given, for which we now need to show that
 1306 $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$. The sequence c_1, c_2, \dots starts at some position 2ℓ within the
 1307 segment $(\mathcal{C}_{2\ell} \xrightarrow{u_{\ell+1}} \xrightarrow{e_{\ell+1}} \mathcal{C}_{2\ell+2} \dots$ for some ℓ), and is followed by $m; \iota$ and the full segment

repeated ad infinitum. The morphism c_n can be any of the arrows of this chain, after an arbitrary number of repetitions of the segment. Graphically, the situation can be depicted like this:



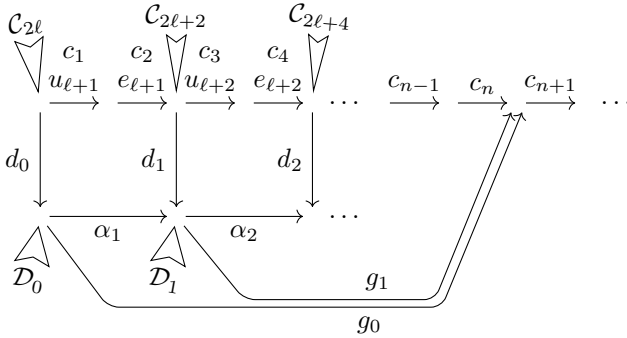
For the purposes of this proof, we need to extend the definition of a direct successor to the step based on $m; \iota$, as follows: First, shifting C_r results in an intermediate condition $\mathcal{H} := C_{r\downarrow m} = \bigwedge_{t \in T} \forall h_t. \mathcal{H}_t$, with each child of C_r being related to its shifted counterparts of \mathcal{H} . As $C_0 \cong C_{r\downarrow m} = \mathcal{H}$ wrt. ι , each child $\forall h_t. \mathcal{H}_t$ is again related to (at least) one child of C_0 , together with an iso ι_t relating the children. Then, each child of C_0 is a successor of the corresponding original child of C_r .

We proceed to construct, as in Theorem 17, a sequence $\forall d_0. \mathcal{D}_0, \forall d_1. \mathcal{D}_1, \dots, \forall d_p. \mathcal{D}_p, \dots, \forall d_q. \mathcal{D}_q$ of children of the universal conditions C_{2i} on the infinite chain of arrows (i.e., the conditions where m and all u_i originate), with each element of the sequence being a (direct) successor of the previous element, such that:

1. After p steps, $0 \leq 2p \leq n$, d_p is an isomorphism,
2. for $k > p$, d_k is an isomorphism as well,
3. after q steps, $p \leq q < \infty$, d_q is pulled forward, resulting in $([e_{q+1}, \dots], \mathcal{D}_q) \in u(P)$.

Afterwards, we can transform that to $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$ (as required by the satisfaction function s) by applying up-to techniques, thereby showing that the segment actually describes an infinite model.

The initial steps are depicted in the following diagram, with the chain $u_{\ell+1}, e_{\ell+1}, \dots, n$, d_0, \mathcal{D}_0 and g_0 given:



The proof objective is now to construct the sequence of all further $\forall d_{k+1}. \mathcal{D}_{k+1}$ and associated α_{k+1} (and, for $k < p$, g_{k+1}).

1. We construct the aforementioned sequence by repeatedly choosing a successor of $\forall d_k. \mathcal{D}_k$ as the next element $\forall d_{k+1}. \mathcal{D}_{k+1}$ of the sequence. We then show that an isomorphism is always obtained after finitely many steps (at most n) by showing that for some p with $2p \leq n$, d_p must be an isomorphism.

If d_k is an isomorphism, we are done, set $p = k$ and we do not need to choose a next element. We will show that $2p \leq n$ later in the proof.

So assume that d_k is not an isomorphism. We consider the next arrow on the chain, which is either u_{k+1} or m :

- Assume the next arrow on the chain results from a tableau step (i.e., u_{k+1}) and neither u_{k+1} nor e_{k+1} equal c_n .

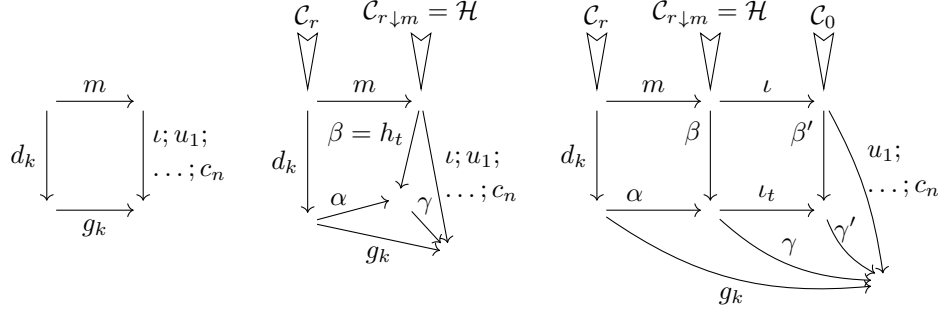
Then construct the representative square and pick the successor as in Theorem 17. Afterwards, start over again, with d_{k+1}, g_{k+1} instead of d_k, g_k . (Since the chain u_{k+2}, \dots, c_n is now two elements shorter than before, this process will not continue endlessly.)

- Consider the case where $c_n = u_{k+1}$ or $c_n = e_{k+1}$ or $n = 0$.

As explained in Theorem 17, in the first case $c_n = u_{k+1}$ is an isomorphism, so $u_{k+1} = d_k; g_k$ can only split into isomorphisms, so d_k must be an isomorphism as well. In the second and third case, we can conclude that $d_{k+1}; g_{k+1} = \text{id}$, since id corresponds to an empty sequence of arrows and again d_{k+1} is an iso.

- Consider the case that the next arrow is $m; \iota$, and it may or may not be c_n .

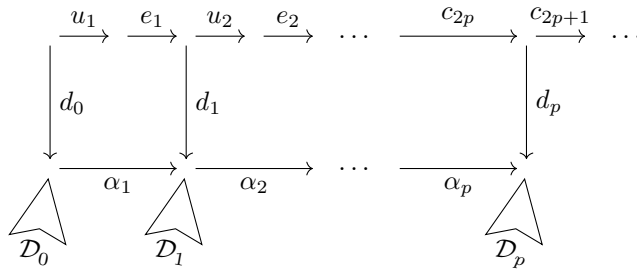
To obtain a successor for $\forall d_k. \mathcal{D}_k$, first reduce the left square below to a representative square (below center), to obtain a corresponding child of \mathcal{H} , which we call $\forall h_t. \mathcal{H}_t$ with $h_t = \beta$. Then, pick a matching isomorphic child $\forall \beta'. \mathcal{H}'_t$ of \mathcal{C}_0 and a relating iso ι_t to obtain the diagram on the right.



Hence the successor of $\forall d_k. \mathcal{D}_k$ is a child $\forall d_{k+1}. \mathcal{D}_{k+1} := \forall \beta'. \mathcal{H}'_t$ of \mathcal{C}_0 , and $g_{k+1} := \gamma' = (\iota_t)^{-1}; \gamma$. We note that the two triangles in the lower right of the diagram above commute: first obviously $\iota_t; \gamma' = \iota_t; (\iota_t)^{-1}; \gamma = \gamma$ and second $\beta'; \gamma' = \beta'; (\iota_t)^{-1}; \gamma = \iota^{-1}; \beta; \gamma = \iota^{-1}; u_1; \dots; c_n = u_1; \dots; c_n$.

If $m; \iota = c_n$, then $d_{k+1}; g_{k+1} = \text{id}$, and hence d_{k+1} must be an isomorphism and we have found the iso d_p . Otherwise, proceed with the remainder of the chain (u_1, e_1, \dots) .

2. Summarizing the situation so far, d_0 has been turned into an isomorphism d_p by now:



To be able to pull an indirect successor of d_p forward, it has to be an isomorphism. See the corresponding part of the proof of Theorem 17 on how to ensure that all

elements $\forall d_{p+1}.\mathcal{D}_{p+1}, \forall d_{p+2}.\mathcal{D}_{p+2}, \dots$ of the sequence do in fact have isomorphisms. (The argument can be applied to the shift over $m; \iota$ as well.)

3. Since d_p is an isomorphism and all its indirect successors are isomorphisms as well, there is an indirect successor that is a direct child of \mathcal{C}_0 and also an iso. As the segment is fair, this means that some successor of it must be pulled forward after finitely many further steps (before reaching \mathcal{C}_r the next time).

By the same reasoning as in Theorem 17, for some $q \geq p$, we reach a condition $\mathcal{C}_t = \bigwedge_{i \in I} \forall f_i.\mathcal{A}_i$ that contains $\forall f_\ell.\mathcal{A}_\ell = \forall d_q.\mathcal{D}_q$, which is an indirect successor of $\forall d_p.\mathcal{D}_p$ and which is pulled forward. We assume that $\mathcal{D}_q = \bigvee_j \exists g_j.\mathcal{B}_j$. Hence we obtain the following tuple in P :

$$([e_{q+1}, u_{q+2}, e_{q+2}, \dots], \bigvee_j \exists g_j. (\mathcal{B}_j \wedge (\bigwedge_{m \neq \ell} \forall f_m.\mathcal{A}_m)_{\downarrow d_q; g_j})) \in P$$

Here $u_{q+1} = f_\ell = d_q$.

We apply our up-to techniques to obtain the originally desired result (which was to satisfy a given $\forall d_0.\mathcal{D}_0$ by showing for a given n and g_0 such that $c_1; \dots; c_n = d_0; g_0$ that $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$ holds).

As before, \mathcal{D}_q is “contained” in the previously obtained path condition, and up-to conjunction removal (Lemma 42) can be used to obtain a tuple with only \mathcal{D}_q :

$$\implies ([e_{q+1}, u_{q+1}, e_{q+2}, \dots], \bigvee_j \exists g_j.\mathcal{B}_j) = ([e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_q) \in u(P)$$

\mathcal{D}_q has been derived from \mathcal{D}_0 by a series of tableau steps (which involve a shift), or $m; \iota$ step (which involves a shift and a condition isomorphism). Hence, for k ranging from q to 0, we apply a series of steps:

■ If $\mathcal{D}_{k+1} = \mathcal{D}_k \downarrow \alpha_{k+1}$ (tableau step), apply up-to shift (Lemma 44) to obtain:

$$\begin{aligned} & ([\alpha_{k+2}; \dots; \alpha_q; e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_{k+1}) \in u(P) \\ \implies & ([\alpha_{k+1}; \alpha_{k+2}; \dots; \alpha_q; e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_k) \in u(P) \end{aligned}$$

■ If \mathcal{D}_{k+1} is the result of a step based on $m; \iota$, we know that $\forall d_k.\mathcal{D}_k$ is a child of \mathcal{C}_r , while $\forall d_{k+1}.\mathcal{D}_{k+1}$ – its successor – is a child of \mathcal{C}_0 .

Now first apply up-to condition isomorphism (Lemma 45), then apply up-to shift (Lemma 44) to undo the shift:

$$\begin{aligned} & ([\alpha_{k+2}; \dots; \alpha_q; e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_{k+1}) \in u(P) \\ \implies & ([\iota; \alpha_{k+2}; \dots; \alpha_q; e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{H}_\iota) \in u(P) \\ \implies & ([\alpha; \iota; \alpha_{k+1}; \dots; \alpha_q; e_{q+1}, u_{q+2}, e_{q+2}, \dots], \mathcal{D}_k) \in u(P) \end{aligned}$$

Note that $\mathcal{D}_k \downarrow \alpha = \mathcal{H}_\iota$.

We eventually obtain a tuple with \mathcal{D}_0 as the condition. Now we use up-to recomposition (Lemma 43) to rewrite the initial part of the chain (see Theorem 17 for details). For the step based on $m; \iota$ we require commutativity of the two triangles that was shown previously. Finally we obtain:

$$([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P) \quad \blacktriangleleft$$

D

 Proofs and Additional Material for §5 (Satisfiability in the General Case)

We now show some straightforward facts on sections in the context of conditions.

► **Lemma 46.** *Let $s: A \rightarrow B$ be a section, where $r: B \rightarrow A$ is a right inverse of s , i.e., $s; r = \text{id}_A$. Then $\forall s.\mathcal{A} \models \mathcal{A}_{\downarrow r} \models \exists s.\mathcal{A}$. However, in general, $\exists s.\mathcal{A} \not\models \mathcal{A}_{\downarrow r} \not\models \forall s.\mathcal{A}$.*

Proof. Note that $\text{RO}(\forall s.\mathcal{A}) = \text{RO}(\exists s.\mathcal{A}) = \text{RO}(\mathcal{A}_{\downarrow r}) = A$ and $\text{RO}(\mathcal{A}) = B$. In the following let $\bar{c} = [c_1, c_2, \dots]$ be a composable sequence of arrows.

■ $(\forall s.\mathcal{A} \models \mathcal{A}_{\downarrow r})$:

$$\begin{aligned}
 & \bar{c} \models \forall s.\mathcal{A} \\
 & \text{(Def. 2 satisfaction)} \iff \forall n. \forall g. (c_1; \dots; c_n = s; g \implies [g, c_{n+1}, \dots] \models \mathcal{A}) \\
 & \text{(choose } g = r; c_1; \dots; c_n) \implies \forall n. (c_1; \dots; c_n = s; r; c_1; \dots; c_n \\
 & \implies [r; c_1; \dots; c_n, c_{n+1}, \dots] \models \mathcal{A}) \\
 & \text{(antecedent is true)} \implies \forall n. [r; c_1; \dots; c_n, c_{n+1}, \dots] \models \mathcal{A} \\
 & \text{(Def. 6 shift)} \iff [c_1; \dots; c_n, c_{n+1}, \dots] \models \mathcal{A}_{\downarrow r} \\
 & \text{(recompose)} \iff \bar{c} \models \mathcal{A}_{\downarrow r}
 \end{aligned}$$

Note that the last step is true since $u_s(\models)$ equals \models (see Section 2.1) and hence satisfaction is preserved by recomposition.

■ $(\mathcal{A}_{\downarrow r} \models \exists s.\mathcal{A})$:

$$\begin{aligned}
 & \bar{c} \models \mathcal{A}_{\downarrow r} \\
 & \text{(Prop. 6 shift)} \iff [r; c_1, c_2, \dots] \models \mathcal{A} \\
 & \text{(ex. intro. using } g = r; c_1) \implies \exists g. (c_1 = s; g \wedge [g, c_2, \dots] \models \mathcal{A}) \\
 & \text{(Def. 2 satisfaction)} \iff \bar{c} \models \exists s.\mathcal{A}
 \end{aligned}$$

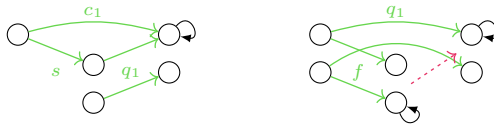
■ $(\exists s.\mathcal{A} \not\models \mathcal{A}_{\downarrow r} \not\models \forall s.\mathcal{A})$:

We show this via a counterexample using the category **Graph_{fin}** of finite graphs and morphisms. Representative squares are given by pushouts.

Here we consider only finite composable sequences, given by a single morphism. Let the following graph morphisms and conditions be given, where the mappings of the morphisms are indicated in green:

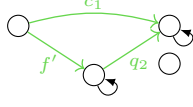
$$\begin{array}{llll}
 c_1 = \begin{array}{c} \text{---} \circ \xrightarrow{\text{green}} \begin{array}{c} \circ \text{---} \circ \\ \circ \end{array} \end{array} & s = \begin{array}{c} \text{---} \circ \xrightarrow{\text{green}} \begin{array}{c} \circ \text{---} \circ \\ \circ \end{array} \end{array} & f = \begin{array}{c} \begin{array}{c} \circ \text{---} \circ \\ \circ \end{array} \xrightarrow{\text{green}} \begin{array}{c} \circ \text{---} \circ \\ \circ \end{array} \end{array} & \mathcal{A} = \forall f.\text{false} \\
 c_2 = \begin{array}{c} \text{---} \circ \xrightarrow{\text{green}} \begin{array}{c} \circ \text{---} \circ \\ \circ \end{array} \end{array} & r = \begin{array}{c} \begin{array}{c} \circ \text{---} \circ \\ \circ \end{array} \xrightarrow{\text{green}} \begin{array}{c} \circ \text{---} \circ \\ \circ \end{array} \end{array} & f' = \begin{array}{c} \begin{array}{c} \circ \text{---} \circ \\ \circ \end{array} \xrightarrow{\text{green}} \begin{array}{c} \circ \text{---} \circ \\ \circ \end{array} \end{array} & \mathcal{A}_{\downarrow r} = \forall f'.\text{false}
 \end{array}$$

■ $c_1 \models \exists s.\mathcal{A}$ because $\exists q_1: c_1 = s; q_1$ and $q_1 \models \forall f.\text{false}$

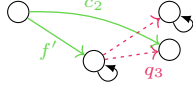


XX:42 Coinductive Techniques for Checking Satisfiability of Generalized Nested Conditions

1438 $\models c_1 \not\models \mathcal{A}_{\downarrow r} = \forall f'. \text{false}$ because $\exists q_2: c_1 = f'; q_2$ but $q_2 \not\models \text{false}$



1439
1440 $\models c_2 \models \mathcal{A}_{\downarrow r}$ because $\nexists q_3: c_2 = f'; q_3$



1441
1442 $\models c_2 \not\models \forall s. \mathcal{A}$ because $\exists q_4: c_2 = s; q_4$ but $q_4 \not\models \forall f. \text{false}$



1443

1444 **► Lemma 30** (Pulling forward sections). *Let $\bigwedge_{i \in I} \forall f_i. \mathcal{A}_i$ be a universal condition and assume*
1445 *that $f_p, p \in I$, is a section, and r_p is a right inverse of f_p (i.e., $f_p; r_p = \text{id}$). Furthermore let*
1446 *$\mathcal{A}_{p \downarrow r_p} = \bigvee_{j \in J} \exists h_j. \mathcal{H}_j$ be the result of shifting the p -th child over the right inverse. Then f_p*
1447 *can be pulled forward:*

$$1448 \quad \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \equiv \bigvee_{j \in J} \exists h_j. \left(\mathcal{H}_j \wedge \left(\bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \right)_{\downarrow h_j} \right)$$

Proof.

$$1449 \quad \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i = \bigwedge_{m \in I \setminus \{p\}} \forall f_m. \mathcal{A}_m \wedge \forall f_p. \mathcal{A}_p$$

$$1450 \quad (*) \equiv \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \wedge \mathcal{A}_{p \downarrow r_p}$$

$$1451 \quad = \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i \wedge \bigvee_{j \in J} \exists h_j. \mathcal{H}_j$$

$$1452 \quad (\text{Lemma 40 and distributivity}) \equiv \bigvee_{j \in J} \exists h_j. \left(\mathcal{H}_j \wedge \bigwedge_{i \in I} (\forall f_i. \mathcal{A}_i)_{\downarrow h_j} \right)$$

1453 $(*)$: from Lemma 46 we know that $\forall f_p. \mathcal{A}_p \models \mathcal{A}_{p \downarrow r_p}$. Hence, since $\mathcal{C} \models \mathcal{D}$ implies $\mathcal{C} \equiv \mathcal{C} \wedge \mathcal{D}$,
1454 we have that $\forall f_p. \mathcal{A}_p \equiv \forall f_p. \mathcal{A}_p \wedge \mathcal{A}_{p \downarrow r_p}$. ◀

1455 **► Theorem 34** (Fair branches are models (general case)). *Let \mathcal{A}_0 be an alternating condition.*
1456 *Let a fixed tableau constructed by the rules of Definition 31 be given. Let $\mathcal{A}_0 \xrightarrow{b_1} \mathcal{A}_1 \xrightarrow{b_2} \mathcal{A}_2 \xrightarrow{b_3} \dots$*
1457 *be a branch of the tableau that is either unextendable and ends with a universal*
1458 *quantification, or is infinite and fair. For such a branch, we define: $P = \{(\bar{b}, \mathcal{A}_i) \mid i \in \mathbb{N}_0, \bar{c} = [b_{i+1}, b_{i+2}, \dots]\} \subseteq \text{Seq} \times \text{Cond}$. Then, $P \subseteq s(u(P))$. (Every open and unextendable*
1459 *or infinite and fair branch in a tableau of Definition 31 corresponds to a model.)*
1460

1461 **Proof.** The general structure of the proof is similar to the proof of Theorem 17. For detailed
1462 explanations, we refer to the proof of the special case, and only highlight the changes that
1463 are necessary for the general case. Note that once a condition contains a section (on the top
1464 level), the branch will always be infinite, since sections are never removed.

1465 Let $\mathcal{A} = \mathcal{C}_0 \xrightarrow{c_1} \mathcal{C}_1 \xrightarrow{c_2} \mathcal{C}_2 \xrightarrow{c_3} \dots$ be a (suffix of a) branch of a tableau that results in a
1466 given $([c_1, c_2, c_3, \dots], \mathcal{A}) \in P$. We show that $([c_1, c_2, c_3, \dots], \mathcal{A}) \in s(u(P))$.

1467 ■ **(\mathcal{A} is existential):**

1468 Exactly as in the special case.

1469 ■ **(\mathcal{A} is universal and contains no sections):**

1470 Assume that $\mathcal{A} = \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i$. The argument is similar to the special case: If \mathcal{A} contains
 1471 no sections, the tableau has no child nodes for \mathcal{A} and the path ends, i.e., we have $([], \mathcal{A})$
 1472 ($[]$ representing id), which is in $s(u(P))$ because there is no arrow g that makes $\text{id} = f_i; g$
 1473 true (if there were such an arrow, then f_i would be a section).

1474 ■ **(\mathcal{A} is universal and has at least one section):**

1475 We now assume that at least one f_i is a section.

1476 For $([c_1, c_2, \dots], \bigwedge_{i \in I} \forall f_i. \mathcal{A}_i) \in s(u(P))$ to hold, by definition of s , for all children $\forall f_i. \mathcal{A}_i$,
 1477 all arrows g and all $n \in \mathbb{N}_0$, we need to show: if $c_1; \dots; c_n = f_i; g$, then $([g, c_{n+1}, \dots], \mathcal{A}_i) \in$
 1478 $u(P)$.

1479 Hence let a particular child $\forall d_0. \mathcal{D}_0 = \forall f_i. \mathcal{A}_i$ to be satisfied, some n , and some g_0 such that
 1480 $c_1; \dots; c_n = d_0; g_0$, be given, for which we need to show that $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$.

1481 Since at least one child is a section, in the next step on the branch, some $\forall f_\ell. \mathcal{A}_\ell$ (f_ℓ section)
 1482 is pulled forward, so the next condition on the path has the shape $\bigvee_j \exists h_j. (\mathcal{H}_j \wedge \dots)$.

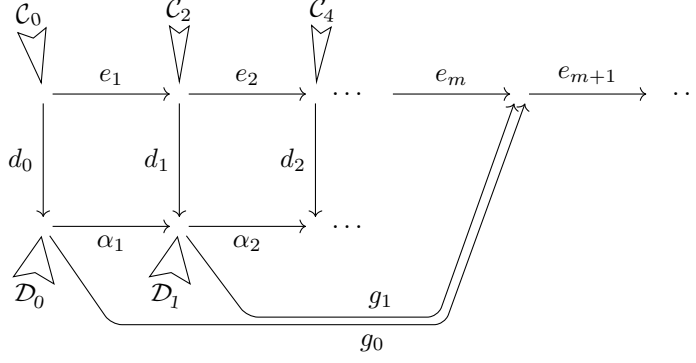
1483 Since the condition is alternating and the tableau rules preserve this property, the sequence
 1484 of c_1, c_2, \dots on the path alternates between id labels (from a universal condition) and
 1485 e_i (from an existential condition). In particular $c_{2i} = e_i$. Without loss of generality
 1486 we can assume that n is even and $c_n = e_{n/2}$. (Otherwise $c_n = \text{id}$ and we know that
 1487 $c_1; \dots; c_{n-1} = d_0; g_0$, where c_{n-1} a morphism corresponding to the existential step. Then
 1488 we can infer $([g_0, c_n, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$ and finally $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$ via
 1489 up-to recomposition.) We set $m = n/2$.

1490 We now claim that there exists a finite sequence $\forall d_0. \mathcal{D}_0, \forall d_1. \mathcal{D}_1, \dots, \forall d_m. \mathcal{D}_m, \dots, \forall d_q. \mathcal{D}_q$
 1491 of children of the universal conditions $\mathcal{C}_0, \mathcal{C}_2, \dots$ on the branch (i.e., every second one),
 1492 with each element of the sequence being a (direct) successor of the previous element, such
 1493 that:

- 1494 1. After m steps, d_m is a section, having a right-inverse r_m ,
- 1495 2. for $k > m$, d_k is a section as well, with a right-inverse r_k such that (d_k, r_k) is a
 1496 retraction successor of (d_m, r_m) .
- 1497 3. after q steps, $m \leq q < \infty$, d_q is pulled forward with r_q (where (d_q, r_q) is a retraction
 1498 successor of (d_m, r_m)), resulting in $([e_{q+1}, \dots], \mathcal{D}_{q \downarrow r_q}) \in u(P)$.

1499 Afterwards, we can transform that to $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$ (as required by the
 1500 satisfaction function s) by applying up-to techniques, thereby showing that the path
 1501 actually describes a model.

1502 Note that $n = 0$ or $n = q$ are possible as well. The initial steps are depicted in the
 1503 following diagram, with the chain $(\text{id}, e_1, (\text{id}, e_2, \dots, n, d_0, \mathcal{D}_0$ and g_0 given (e_1, e_2, \dots
 1504 are labels of existential steps and hence correspond to the morphisms h_j of the tableau
 1505 construction rules):



1506

1507

1508

The proof objective is now to construct the sequence of all further $\forall d_{k+1}.\mathcal{D}_{k+1}$, the associated α_{k+1} , and g_{k+1} (for $k < m$) or r_k (for $k \geq m$).

1509

1510

1511

1512

1513

1514

1515

1516

1517

1518

1519

1520

1521

1522

1523

1524

1525

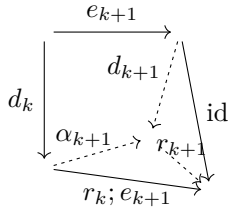
1526

1527

1528

1529

1. We construct the aforementioned sequence by repeatedly choosing a successor of $\forall d_k.\mathcal{D}_k$ as the next element ($\forall d_{k+1}.\mathcal{D}_{k+1}$) of the sequence, then show that d_m is a section. See the proof of the special case, replacing “isomorphism” with “section”. Also instead of stopping when a section is first obtained, continue until d_m is reached. Additionally, $d_m; g_m = \text{id}$ (with id being the empty sequence of the c_i), so g_m is a right-inverse of d_m , which we also choose as the starting point of the sequence of retractions (i.e., $r_m := g_m$). Note that in the special case, we considered the first occurrence of an isomorphism successor as the p -th step (with p possibly being less than m , equivalently $2p \leq n$). Now we consider exactly the first m steps, even if a section is obtained earlier already. As the general pull-forward rule does not “consume” successors (unlike the special rule) even if they are pulled forward earlier, we can still ensure that after m steps, we still have a successor that can be pulled forward.
2. Summarizing the situation so far, d_0 has been turned into a section d_m and a chosen right-inverse r_m (such that $d_m; r_m = \text{id}$) by now. We use the retraction successor relation as described in Definition 32 to obtain, for $n \leq k \leq q$, the sequence of subsequent $\alpha_{k+1}, d_{k+1}, r_{k+1}$ (named α, β, r_β , respectively, in Definition 32). Note that Lemma 7 guarantees the existence of a section d_{k+1} in the next step. Furthermore we also obtain retraction successors, namely retractions r_k, r_{k+1} such that $\alpha_{k+1}; r_{k+1} = r_k; e_{k+1}$ (see diagram below).



1530

1531

1532

1533

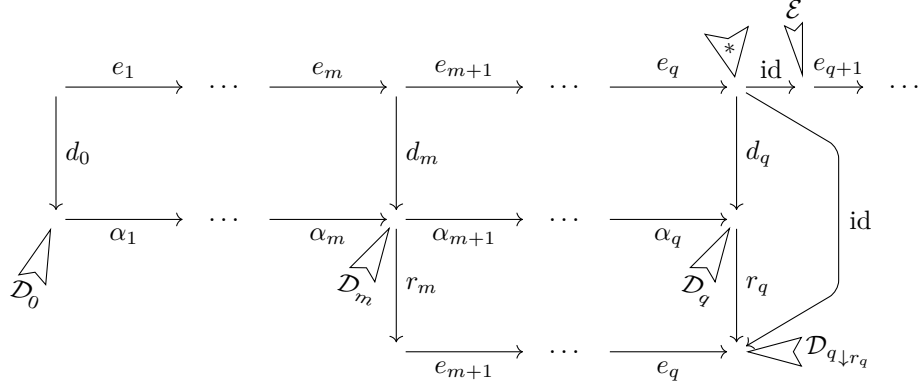
1534

1535

1536

3. Since d_m is a section with retraction r_m , one of its retraction successor must eventually pulled forward if we make an appropriate choice of representative squares above. From the arguments above, we know that the path is infinite and the fairness constraint guarantees the existence of a finite q . We end up with a situation depicted below, the condition \mathcal{C}_q at the current node being marked by $*$. Note that the upper row of squares commutes by construction, in

particular, $d_i; \alpha_{i+1} = e_{i+1}; d_{i+1}$, while this is not necessarily true for the lower row of squares.



Now let q be the step at which a successor d_q of $\forall d_m. \mathcal{D}_m$ is finally pulled forward using inverse r_q . Let $\mathcal{C}_q = \bigwedge_{\ell \neq t} \forall f_\ell. \mathcal{A}_\ell$ and $\forall d_q. \mathcal{D}_q = \forall f_t. \mathcal{A}_t$ for some index t . Assume that $(\mathcal{D}_q)_{\downarrow r_q} = \bigvee_j \exists h_j. \mathcal{H}_j$, then the next step in the tableau is:

$$\mathcal{C}_q = \left(\bigwedge_{\ell \neq t} \forall f_\ell. \mathcal{A}_\ell \right) \wedge \forall d_q. \mathcal{D}_q \xrightarrow{\text{id}} \overbrace{\bigvee_j \exists h_j. \left(\mathcal{H}_j \wedge \left(\bigwedge_i \forall f_i. \mathcal{A}_i \right)_{\downarrow h_j} \right)}^{\mathcal{E}}$$

$\mathcal{D}_{q \downarrow r_q}$

As a result, we have:

$$([e_{q+1}, \text{id}, e_{q+2}, \dots], \mathcal{E}) \in P$$

We now apply our up-to techniques to obtain the originally desired result (which was to satisfy a given $\forall d_0. \mathcal{D}_0$ by showing for a given n and g_0 such that $c_1; \dots; c_n = d_0; g_0$ that $([g_0, c_{n+1}, \dots], \mathcal{D}_0) \in u(P)$ holds).

Observe that $\mathcal{D}_{q \downarrow r_q}$ is “contained” in \mathcal{E} , in the sense that the path condition \mathcal{E} is a (nested) conjunction of $\mathcal{D}_{q \downarrow r_q}$ and additional conditions. First, we apply up-to conjunction removal (Lemma 42) to turn our condition \mathcal{E} (“ $\mathcal{D}_{q \downarrow r_q}$ with conjunctions”) into just $\mathcal{D}_{q \downarrow r_q}$:

$$\Rightarrow ([e_{q+1}, \text{id}, e_{q+2}, \dots], \bigvee_j \exists h_j. \mathcal{H}_j) = ([e_{q+1}, \text{id}, e_{q+2}, \dots], \mathcal{D}_{q \downarrow r_q}) \in u(P)$$

Since $\mathcal{D}_q = \mathcal{D}_{q-1 \downarrow \alpha_q} = \dots = \mathcal{D}_{0 \downarrow \alpha_1 \downarrow \dots \downarrow \alpha_q}$, we can now apply a series of up-to shift operations (Lemma 44) to obtain a tuple with \mathcal{D}_0 as the condition:

$$\begin{aligned} & ([e_{q+1}, \dots], \mathcal{D}_{q \downarrow r_q}) \in u(P) \\ \Rightarrow & ([r_q; e_{q+1}, \dots], \mathcal{D}_q) = \\ & ([r_q; e_{q+1}, \dots], \mathcal{D}_{0 \downarrow \alpha_1 \downarrow \dots \downarrow \alpha_q}) \in u(P) \\ \Rightarrow & ([\alpha_q; r_q; e_{q+1}, \dots], \mathcal{D}_{0 \downarrow \alpha_1 \downarrow \dots \downarrow \alpha_{q-1}}) \in u(P) \\ \Rightarrow & \dots \Rightarrow ([\alpha_1; \dots; \alpha_q; r_q; e_{q+1}, \dots], \mathcal{D}_0) \in u(P) \end{aligned}$$

1560 Now apply previous results to rewrite the initial part of the chain. Remember that we
 1561 obtain $\alpha_{k+1}; r_{k+1} = r_k; e_{k+1}$ from the choice of retraction successors.
 1562 Hence

$$1563 \quad \alpha_{n+1}; \dots; \alpha_q; r_q = \alpha_{n+1}; \dots; \alpha_{q-1}; r_{q-1}; e_q = \dots = r_m; e_{m+1}; \dots; e_q,$$

1564 therefore

$$1565 \quad ([\alpha_1; \dots; \alpha_m; r_m; e_{m+1}; \dots; e_q; e_{q+1}, \dots], \mathcal{D}_0) \in u(P)$$

1566 Next, we exploit the way in which the g_i were constructed (with the last one being
 1567 $g_m = r_m$). In particular $\alpha_{i+1}; g_{i+1} = g_i$, hence:

$$1568 \quad \alpha_1; \dots; \alpha_n; r_m = \alpha_1; \dots; \alpha_m; g_m = \alpha_1; \dots; \alpha_{m-1}; g_{m-1} = \dots = g_0$$

1569 As a result, we have

$$1570 \quad ([g_0; e_{m+1}; \dots; e_q; e_{q+1}, \dots], \mathcal{D}_0) \in u(P)$$

1571 With up-to recomposition (Lemma 43), we can adjust this to

$$1572 \quad ([g_0, \text{id}, e_{m+1}, \text{id}, \dots, e_q, \text{id}, e_{q+1}, \dots], \mathcal{D}_0) \in u(P)$$

1573 which corresponds, as desired, to

$$1574 \quad ([g_0, c_{n+1}, c_n, \dots], \mathcal{D}_0) \in u(P). \quad \blacktriangleleft$$