A fully local proof system for first-order logic

Victoria Barrett
Inria & LIX, Institut Polytechnique Paris, France

Kaustuv Chaudhuri
Inria & LIX, Institut Polytechnique Paris, France

Abstract

Checking an instance of an inference rule in a proof system can involve computations of arbitrary complexity owing to: the shape of inference rule, the assumed structural properties on the judgemental structures, and side conditions. To more precisely account for the complexity of checking a proof, *deep-inference* calculi have been proposed for various propositional logics that reduce structural rules to *atomic form* and logical rules to *linear form*, so every inference rule can be checked by examining a constant portion of the formula tree; such systems are called *local*. This paper provides a local proof system for *first-order* (classical) logic by conservatively extending the existing atomic system SKSq with rules to localise the term operations associated with (co)contraction, (co)weakening, identity, cut, (co)instantiation, and (co)vacuousness. This is achieved by extending the syntax of derivations with explicit subsitutions; extending terms with formal notations for arities, (co)duplication, and (co)deletion; and allowing inferences inside atoms and within the quantifier prefixes.

2012 ACM Subject Classification Theory of computation → Proof theory

Keywords and phrases Deep-Inference, Open Deduction, First-Order Logic, Classical Logic

Digital Object Identifier 10.4230/LIPIcs.CSL.2027.94

Funding *Victoria Barrett* : Partially funded by the *Action Exploratoire* IMPROOF.

1 Introduction

Here are some standard inference rules of the sequent calculus (e.g., G3c from [18]):

$$\mathsf{id}\,\frac{}{\vdash A, A^\perp} \quad \mathsf{cut}\,\frac{\vdash \Gamma, A \quad \vdash \Delta, A^\perp}{\vdash \Gamma, \Delta} \quad \mathsf{ct}\,\frac{\vdash \Gamma, A, A}{\vdash \Gamma, A} \quad \mathsf{wk}\,\frac{\vdash \Gamma}{\vdash \Gamma, A} \quad \exists\,\frac{\vdash \Gamma, [t/x]\,A}{\vdash \Gamma, \exists x.\,A} \quad \forall\,\frac{\vdash \Gamma, [u/x]\,A}{\vdash \Gamma, \forall x.\,A}\, u\#(\Gamma, A)$$

In this paper we are concerned with *locality* of checking a purported instance of a rule: *how much of the sequent must be examined to ensure that the rule instance is correct?* A fully local system would be one where every instance would be checkable by examining only a *constant* portion of the sequent. The sequent calculus enjoys a kind of locality: each rule instance can be checked independently of every other rule instance. Nevertheless, the rules themselves exhibit several non-local features:

- The id and cut rules have multiple (dualised) occurrences of a formula, so the checker must verify that these occurrences are the same (or dual) formula. Checking cut additionally requires verifying that the conclusion context is correctly split.
- The ct rule makes a (deep) copy of a formula, while the wk rule has an implicit premise that A is indeed a formula.
- The $\exists$ rule in a conclusion-upwards direction requires verifying that t is a well-formed term, and the correct replacement of potentially several occurrences of x in A with t.
- The $\forall$ rule has a side condition that the *eigenvariable* u is not free in Γ, A.

Can the sequent calculus be made local? Some steps in this direction are well known: the id rule can be restricted to those cases where A is an atomic formula (*identity expansion*); and atoms are potentially less costly to compare than arbitrary formulas; the cut rule can be shown to be admissible and removed (*cut-elimination*); and structural rules such as wk

can be performed lazily. However, the sequent calculus has limitations: the ct rule, for instance, cannot be reduced to atomic form (without additional bureaucratic cuts). Worse, cut-elimination is computationally expensive, but it is unreasonable to define locality only for cut-free systems. Indeed, one of the goals of locality is to expose finer grained normalisation procedures [10, 15, 12, 3].

Further improvements in locality can be achieved by moving to a *deep-inference* proof system such as the *calculus of structures* [9, 7], where rule instances are allowed more freely within subformulas, which in turn allows structural rules such as ct to be applied on the smallest components of a formula. The system SKS [6] for classical propositional logic has the following remarkable property: *every inference rule is fully local.* This is achieved in two steps: first, the logical inference rules are written in a *linear* form [4], where the only actions performed are modifications of a finite number of nodes at the root of a formula tree, without altering the form or the count of the subformula occurrences. This is clearly seen in the *switch* s and *medial* m rules:

$$\mathsf{s}\,\frac{(A \wedge B) \vee C}{A \vee (B \wedge C)} \qquad \mathsf{m}\,\frac{(A \wedge C) \vee (B \wedge D)}{(A \vee B) \wedge (C \vee D)}$$

where the verifier only needs to “rewire” the subformulas $A, \ldots, D$ without looking inside them. More importantly, the structural rules of SKS are made *atomic*; in particular:

$$\mathsf{ai}\,\frac{\top}{\mathsf{a} \vee \mathsf{a}^{\perp}} \qquad \mathsf{acut}\,\frac{\mathsf{a} \wedge \mathsf{a}^{\perp}}{\perp} \qquad \mathsf{ac}\,\frac{\mathsf{a} \vee \mathsf{a}}{\mathsf{a}} \qquad \mathsf{aw}\,\frac{\perp}{\mathsf{a}}$$

Each instance of these rules can be checked by examining a constant number of constant sized objects. Furthermore, the non-atomic versions of these rules can be derived using the atomic versions with essentially no increase in size, by means of the medial rule m. Related local systems also exist for propositional linear logic [16] and intuitionistic logic [17].

In this paper we ask if a similar kind of result can be obtained for *first-order* (classical) logic. This problem was explicitly left open in [6, Chapter 3] and has remained so in the 22 years since. One reason for the lack of progress might be that while deep-inference calculi for first-order logic can be reduced to atomic form, such as in the system SKSq in [6], this reduction is not sufficient for controlling the costs in first-order logic. This is immediately apparent in the first-order version of the ai and ac rules:

$$\mathsf{ai}\,\frac{\top}{\mathsf{p}(t_1, \ldots, t_n) \vee \mathsf{p}(t_1, \ldots, t_n)^{\perp}} \qquad \mathsf{ac}\,\frac{\mathsf{p}(t_1, \ldots, t_n) \vee \mathsf{p}(t_1, \ldots, t_n)}{\mathsf{p}(t_1, \ldots, t_n)}$$

To check these rules, the checker has to verify that the two (or three) occurrences of each term t_i are the same, which can have arbitrary size. Note: this issue cannot be finessed by implementation techniques such as hash-consing, because it still requires the several occurrences of each of the t_i in the rule instance to be given the same global identifier (i.e., hash code), which still requires full traversal. Moreover, the problem of non-locality of the quantifier rules (such as instantiation for $\exists$) remains entirely unsolved (and remains so even with approaches involving the ε-calculus [**?**], for example).

Our approach to these challenges begins by generalising the notion of derivation from just an operation on (sub)formulas to operations on (sub)terms; indeed, we design an inference system that operates on partial applications of terms and predicates. Our approach can therefore be seen as belonging to the broad class of *subatomic proof systems* [2, 3] that interpret atoms as connectives. The only objects in our syntax with constant cost (de)duplication or deletion will be variables, predicate symbols, and function symbols; entire terms and atoms will need to be (de)duplicated or deleted piece by piece. In the process we will extend the syntax of terms with operators denoting (co)duplication and (co)deletion, each of which will get inference rules with the *medial rule shape*.

Next, we tackle instantiation of $\exists$ by witness terms. Our approach is to construct the witness term piece by piece, as a sequence of instantiations, each step having constant cost. For instance, to instantiate $\exists x. A$ to $[\mathsf{f}(\mathsf{c}, u)/x]\, A$, we first instantiate x to $g(y)$, then g to $\mathsf{f}(z)$, then y to u, and finally z to c. The intermediate stages of this instantiation process are represented using *explicit substitutions* [1], while the steps of instantiation are written as inference rules within the terms and the substitutions in the style of *open deduction* [3]. While explicit substitutions in deep-inference are not new, there is a long standing open question of designing a symmetric proof system with explicit substitutions that can handle duality; we solve this in this paper without side-stepping it with negation-free subatomic logic as in [4]. We also employ a novel notion of *inference within the quantifier prefix*, which is a new member of the family of (vertical) compositions by inference in open deduction.

Next, we show how to give a local treatment to adding a vacuous quantifier for a variable that does not occur free in a formula. This is the essence of the eigenvariable side condition, which is obviously not a constant cost check, but in SKSq the addition and removal of vacuous quantifiers was treated as part of the ambient equational theory on formulas. We repurpose the new term unit $\blacktriangle$ that was introduced for coweakening, and then design an inference system that allows $\blacktriangle$ as a stand-in for a vacuously quantified term. In the conclusion-upwards direction, a substitution of $\blacktriangle$ for x can be interpreted as a constant cost vacuous quantification, which remains sound because the only rules available for $\blacktriangle$ in the premise-downwards direction would rewrite the formula to $\top$ if the quantification was not indeed vacuous. This interplay of coweakening and vacuousness was heretofore unknown in deep inference and only becomes visible when moving contraction and weakening into the terms.

Finally, we show how to reduce atomic identity ai (and its dual, atomic cut) to a local form. The essence of these rules is to compare terms structurally, which we achieve by adding an intensional term equality $\mathrel{\mathring{=}}$ predicate (and its dual $\mathrel{\mathring{\neq}}$) with local rules (similar to [8]). The key novelty here is that we must handle the extended term syntax for (co)duplication and (co)deletion mentioned above, which do not behave identically on both sides of an identity or cut. Our approach therefore generalises the notion of duality to terms such that it remains an identity on ordinary terms but dualises for the extended terms. Furthermore, since terms and predicates may be partially applied, we further generalise these operations to higher *arities*, which is a minimal type system sufficient to localise the rules.

The structure of the paper is as follows: in Section 2 we give the preliminaries for open deduction and first-order logic; in Sections 3–5 we perform the steps described above to make the inference rules contraction, weakening, instantiation, vacuous quantification, and identity local, culminating in the proof system SKSqt; in Section 6 we state the main contribution of this paper, that SKSqt can prove any theorem of first-order logic and that it does not prove any unwanted theorem. The proofs of these statements are given in Appendices C and D.

2 Background: Open Deduction for First-Order Logic

This section sketches a formulation of first-order classical logic in *deep-inference*. The first deep-inference calculus, the *calculus of structures*, resulted from observing that inference rules can be applied in formula contexts: if $\frac{A}{B}$ is a valid inference in the system, then so is $\frac{C\{A\}}{C\{B\}}$ where $C\{\ \}$ is a (negation-normalised) formula with a single occurrence of the hole $\{\ \}$, and $C\{F\}$ represents the formula resulting from replacing that hole with F. Indeed, given a derivation (i.e., a sequence of unary inferences) Φ with premise A and conclusion B, there is a derivation $C\{\Phi\}$ with premise $C\{A\}$ and conclusion $C\{B\}$ that results from placing each formula in Φ within $C\{\,\}$. In other words, derivations may be substituted for the hole in a formula context, blurring the distinction between formula and derivation.

Open deduction further blurs the distinction between formula and derivation by observing that two derivations Φ and Θ may be *horizontally* combined with logical connectives such as $\wedge$, yielding a derivation from their $\wedge$-ed premises to their $\wedge$-ed conclusion. This yields a parallelised syntax for deep-inference that abstracts out the bureaucracy of interleaved unary rules in the calculus of structures.

▶ **Definition 1** (variables, terms, and atoms). *Let $\mathcal{V}$ be an infinite set of* **variables**, *and for every $i \in N$, let $\mathcal{C}_i$ be a non-empty*[1] *set of* **constants** *of arity i, and $\mathcal{P}_i$ be a set of* **predicates** *of arity i. For every $i \in N$, the set of formal dual predicates $\overline{\mathcal{P}_i}$ is the set $\{\overline{\mathsf{p}} \mid \mathsf{p} \in \mathcal{P}_i\}$, and the set of* **predicate literals** *$\mathcal{L}_i$ is $\mathcal{P}_i \uplus \overline{\mathcal{P}_i}$. The set of* **terms**, *$\mathcal{T}$, and the set of* **atoms**, *$\mathcal{A}$, are given by the grammars:*

$$\mathcal{T} \ ::= \ \mathcal{V} \mid \mathcal{C}_i(\underbrace{\mathcal{T},\ldots,\mathcal{T}}_{i}) \qquad\qquad \mathcal{A} \ ::= \ \mathcal{L}_i(\underbrace{\mathcal{T},\ldots,\mathcal{T}}_{i})$$

with the convention that the parentheses are omitted for nullary constants and predicates. We use sans-serif letters $\mathsf{c}, \mathsf{d}, \mathsf{f}, \mathsf{g}$ to denote constants, p, q, r to denote literals, and c to denote a constant or a variable. We use the notation $\vec{t}$ to denote the argument list $(t_1, \ldots, t_i)$ for constants and predicates, so we depict applications schematically as $\mathsf{f}\vec{t}$ or $\mathsf{p}\vec{t}$.

▶ **Definition 2** (derivations). *The set of* **derivations**, *$\mathcal{D}$, in open deduction is given by the following grammar:*

$$\begin{array}{rcll} \mathcal{D} & ::= & \bot \mid \top \mid \mathcal{A} & \textit{units and atoms} \\ & \mid & \mathcal{D} \vee \mathcal{D} \mid \mathcal{D} \wedge \mathcal{D} & \textit{(horizontal) composition by connective} \\ & \mid & \exists \mathcal{V}.\, \mathcal{D} \mid \forall \mathcal{V}.\, \mathcal{D} & \textit{quantification} \\ & \mid & \dfrac{\mathcal{D}}{\mathcal{D}} & \textit{(vertical) composition by inference} \end{array}$$

The set of **formulae** *is the subset $\mathcal{F} \subseteq \mathcal{D}$ consisting of all those derivations constructed without using composition by inference. We use uppercase Greek letters Φ, Ψ, Θ, etc. to denote derivations, and $A, B, \ldots, F$ to denote formulae. In the quantified derivations $\exists x.\, \Phi$ and $\forall x.\, \Phi$, the scope of x extends as far as possible, i.e., the quantifiers have least precedence. The notation $[t/x]\,\Phi$ stands for the (capture-avoiding)* **substitution** *of the term t for the variable x in the derivation Φ.*

▶ **Definition 3** (premises and conclusions). *The* **premise** *and* **conclusion** *of derivations are given, respectively, by two maps $\mathsf{pr}, \mathsf{cn} : \mathcal{D} \to \mathcal{F}$ as follows:*

- *For every $A \in \{\bot, \top\} \cup \mathcal{A}$, $\mathsf{pr}(A) = \mathsf{cn}(A) = A$.*
- *If $* \in \{\vee, \wedge\}$, then $\mathsf{pr}(\Phi * \Theta) = \mathsf{pr}(\Phi) * \mathsf{pr}(\Theta)$ and $\mathsf{cn}(\Phi * \Theta) = \mathsf{cn}(\Phi) * \mathsf{cn}(\Theta)$.*
- *If $\mathsf{Q} \in \{\exists, \forall\}$ then $\mathsf{pr}(\mathsf{Q}x.\, \Phi) = \mathsf{Q}x.\, \mathsf{pr}(\Phi)$ and $\mathsf{cn}(\mathsf{Q}x.\, \Phi) = \mathsf{Q}x.\, \mathsf{cn}(\Phi)$.*
- $\mathsf{pr}\left(\dfrac{\Phi}{\Theta}\right) = \mathsf{pr}(\Theta)$ *and* $\mathsf{cn}\left(\dfrac{\Phi}{\Theta}\right) = \mathsf{cn}(\Phi)$.

▶ **Definition 4** (duals). *The negation map $\overline{(\cdot)} : \mathcal{F} \to \mathcal{F}$ gives the De Morgan* **dual** *of every formula; it is involutive (i.e., $\overline{\overline{A}} = A$) and has the following generating equalities: $\overline{\mathsf{p}\vec{t}} = \overline{\mathsf{p}}\vec{t}$, $\overline{A \vee B} = \overline{A} \wedge \overline{B}$, $\overline{\bot} = \top$, and $\overline{\exists x.\, A} = \forall x.\, \overline{A}$.*

[1] The constant sets are non-empty because our logic is classical.

$$
\begin{array}{llll}
\mathsf{ai}{\downarrow}\,\dfrac{\top}{\mathsf{p}\vec{t} \vee \bar{\mathsf{p}}\vec{t}} & \mathsf{ai}{\uparrow}\,\dfrac{\mathsf{p}\vec{t} \wedge \bar{\mathsf{p}}\vec{t}}{\bot} & \mathsf{n}{\downarrow}\,\dfrac{[t/x]\,A}{\exists x.\,A} & \mathsf{n}{\uparrow}\,\dfrac{\forall x.\,A}{[t/x]\,A} \\[2ex]
\mathsf{ac}{\downarrow}\,\dfrac{\mathsf{p}\vec{t} \vee \mathsf{p}\vec{t}}{\mathsf{p}\vec{t}} & \mathsf{ac}{\uparrow}\,\dfrac{\mathsf{p}\vec{t}}{\mathsf{p}\vec{t} \wedge \mathsf{p}\vec{t}} & \mathsf{qs}{\downarrow}\,\dfrac{\forall x.\,(A \vee B)}{(\forall x.\,A) \vee (\exists x.\,B)} & \mathsf{qs}{\uparrow}\,\dfrac{(\exists x.\,A) \wedge (\forall x.\,B)}{\exists x.\,(A \wedge B)} \\[2ex]
\mathsf{aw}{\downarrow}\,\dfrac{\bot}{\mathsf{p}\vec{t}} & \mathsf{aw}{\uparrow}\,\dfrac{\mathsf{p}\vec{t}}{\top} & \mathsf{m}^{\vee}_{\exists}\,\dfrac{(\exists x.\,A) \vee (\exists x.\,B)}{\exists x.\,(A \vee B)} & \mathsf{m}^{\forall}_{\wedge}\,\dfrac{\forall x.\,(A \wedge B)}{(\forall x.\,A) \wedge (\forall x.\,B)} \\[2ex]
\mathsf{av}{\downarrow}\,\dfrac{\exists x.\,\mathsf{p}\vec{t}}{\mathsf{p}\vec{t}} & \mathsf{av}{\uparrow}\,\dfrac{\mathsf{p}\vec{t}}{\forall x.\,\mathsf{p}\vec{t}} & \mathsf{m}^{\vee}_{\forall}\,\dfrac{(\forall x.\,A) \vee (\forall x.\,B)}{\forall x.\,(A \vee B)} & \mathsf{m}^{\exists}_{\wedge}\,\dfrac{\exists x.\,(A \wedge B)}{(\exists x.\,A) \wedge (\exists x.\,B)} \\[2ex]
\mathsf{s}\,\dfrac{A \wedge (B \vee C)}{(A \wedge B) \vee C} & & \mathsf{m}^{\exists}_{\vee}\,\dfrac{\exists x.\,(A \vee B)}{(\exists x.\,A) \vee (\exists x.\,B)} & \mathsf{m}^{\wedge}_{\forall}\,\dfrac{(\forall x.\,A) \wedge (\forall x.\,B)}{\forall x.\,(A \wedge B)} \\[2ex]
\mathsf{m}\,\dfrac{(A \wedge B) \vee (C \wedge D)}{(A \vee C) \wedge (B \vee D)} & & \mathsf{m}^{\exists}_{\forall}\,\dfrac{\exists y.\,\forall x.\,A}{\forall x.\,\exists y.\,A} &
\end{array}
$$

$$
\begin{array}{ll}
A * B \equiv B * A \qquad A * (B * C) \equiv (A * B) * C & \text{for } * \in \{\vee, \wedge\} \\
A \wedge \top \equiv A \vee \bot \equiv A \qquad \top \vee \top \equiv \top \qquad \bot \wedge \bot \equiv \bot \qquad \forall x.\,\top \equiv \top \qquad \exists x.\,\bot \equiv \bot & \\
\forall x.\,\forall y.\,A \equiv \forall y.\,\forall x.\,A \qquad \exists x.\,\exists y.\,A \equiv \exists y.\,\exists x.\,A \qquad \mathsf{Q}x.\,A \equiv \mathsf{Q}y.\,[y/x]\,A & \text{for } \mathsf{Q} \in \{\exists, \forall\}
\end{array}
$$

Figure 1 The atomic symmetric proof system SKSq for classical first-order logic with rule schemes above the dotted line and the generators for the equational theory below. In the atomic rules in the top left corner, $\mathsf{p} \in \mathcal{L}_i$. In the rules $\mathsf{av}{\downarrow}$ and $\mathsf{av}{\uparrow}$, x must not be free in $\vec{t}$.

▶ **Definition 5** (proof system). *A* **proof system** S *is a set of inference rule schemes of the form* $\rho\,\dfrac{A}{B}$ *where A and B are formulae and ρ is a* name *given to that inference rule scheme. A derivation Φ is a* **derivation in** S *if for every instance of composition by inference* $\dfrac{\Xi}{\Psi}$ *in the construction of Φ,* $\rho\,\dfrac{\mathsf{cn}(\Xi)}{\mathsf{pr}(\Psi)}$*, is in* S*. We write* $\begin{array}{c}A\\ \Phi\,\|\,\mathsf{S}\\ B\end{array}$ *to denote a derivation Φ with premise A, conclusion B where all inference rules used in Φ are in the proof system* S*; we elide Φ and* S *unless relevant.*

▶ **Definition 6.** *An* **equational theory** *is a congruent equivalence relation $\equiv\, : \mathcal{F} \times \mathcal{F}$, often specified as being generated by a set of basic equivalences of formulae. We can extend a proof system* S *with an equational theory $\equiv$ by adding the inference rule scheme* $\equiv\dfrac{A}{B}$ *if $A \equiv B$. This rule is invertible and we will use the notation* $\begin{array}{c}A\\ \vdots\\ B\end{array}$ *to stand for this rule scheme when the equational theory is understood from context.*

We will now give a variant of the proof system SKSq from [6] for first-order logic, reformulated using open deduction instead of the calculus of structures.

▶ **Definition 7.** *The proof system* SKSq *[6] is given by the equational theory and inference rules in Figure 1.*[2] *The first eight rules in the top left corner all related to atoms, and their names are prefixed with* a*.*

One major difference between the system as presented in Figure 1 and the similarly named system in [6] is that the latter system has vacuous quantification as part of the equational

[2] An example derivation of the drinker's formula in SKSq is given in Appendix A.

theory, i.e., it assumes that $\forall x. A \equiv \exists x. A \equiv A$ if x is not free in A. We treat vacuousness in a more fine-grained manner by only including two atomic and directional versions of the rules, $\mathsf{av}\!\downarrow$ and $\mathsf{av}\!\uparrow$, and including only vacuous quantification over units in the equational theory. We add medial rules $\mathsf{m}^{\wedge}_{\forall}$, $\mathsf{m}^{\exists}_{\vee}$, and $\mathsf{m}^{\exists}_{\forall}$ to bridge the gap. Our version is motivated by Section 4 where vacuous quantification is made local. Nevertheless, our version is strongly similar to that of [6]. Some inference rules have a direction up or down, indicated by $\uparrow$ and $\downarrow$ in their names, and if $\rho\!\downarrow \dfrac{A}{B}$ is an inference rule then so is $\rho\!\uparrow \dfrac{\bar{B}}{\bar{A}}$.

▶ **Definition 8.** *A rule $\rho\!\downarrow$ is a* **down-rule** *and a rule $\rho\!\uparrow$ is an* **up-rule**. *The set of all up-rules is the* **up-fragment** *of* SKSq *and the proof system* KSq *is obtained from* SKSq *by removing the up-fragment.*

In the sequent calculus, id and cut can be seen to be dual: the former introduces a dual pair of of formulae downwards in a derivation and the latter upwards. This duality is obscured in their presentation by the asymmetry of sequent calculus proofs as trees with a single conclusion and many premises. Deep-inference proofs have a single premise and conclusion, restoring this top-down symmetry. And, because $\rho\!\downarrow$ and $\rho\!\uparrow$ are both in SKSq for all ρ, entire SKSq-derivations can be dualised, with proofs $\begin{array}{c}\top\\ \Phi\,\Vert\,\mathsf{SKSq}\\ A\end{array}$ becoming refutations $\begin{array}{c}\bar{A}\\ \bar{\Phi}\,\Vert\,\mathsf{SKSq}\\ \bot\end{array}$. This symmetry allows for a lot of freedom in composing derivations and defining normalisation procedures.[3] The entire up-fragment can be seen as corresponding to the cut rule in a sequent system, and cut-elimination in deep-inference takes the form of showing the whole up-fragment to be admissible (see [13] for the proof):

▶ **Theorem 9** (cut elimination for SKSq). *For every $A \in \mathcal{F}$, if* $\begin{array}{c}\top\\ \Vert\,\mathsf{SKSq}\\ A\end{array}$, *then* $\begin{array}{c}\top\\ \Vert\,\mathsf{KSq}\\ A\end{array}$. ◀

3 Localising Contraction and Weakening

The system SKSq in Figure 1 has only atomic versions of the structural rules of identity ($\mathsf{ai}\!\downarrow$), cut ($\mathsf{ai}\!\uparrow$), (co)contraction ($\mathsf{ac}\!\downarrow$ and $\mathsf{ac}\!\uparrow$), (co)weakening ($\mathsf{aw}\!\downarrow$ and $\mathsf{aw}\!\uparrow$), and (co)vacuous quantification ($\mathsf{av}\!\downarrow$ and $\mathsf{av}\!\uparrow$). However, since every one of these rules can mention predicate literals of arbitrary arity and term arguments of arbitrary size, the rules cannot be checked with constant cost. We will decompose these rules into more fine-grained and local notions, but in order to do that we need to depart from the ordinary syntax of first-order terms and predicates to allow inference rules to operate within the atoms and terms.

3.1 Localising Contraction

In the propositional system SKS [6], the general contraction rule $\mathsf{c}\!\downarrow \dfrac{A \vee A}{A}$, where A is an arbitrary formula, is decomposed to its atomic form $\mathsf{ac}\!\downarrow$ by the *medial* rule m (for $\wedge$ the main connective of A) and associativity and commutativity of disjunction (for $\vee$), as shown in the first two rules below:

[3] A little bit of this can be seen in the proof of Theorem 28 in Appendix D, in particular case $\mathsf{gtr}\!\downarrow/(\mathsf{i}^{\mathring{=}}\!\downarrow)$ in Figure 6 and case $\mathsf{gtc}\!\downarrow/\mathsf{ai}'\!\downarrow$ in Figure 8.

$$\mathsf{m}\,\frac{(B \wedge C) \vee (B \wedge C)}{(B \vee B) \wedge (C \vee C)} \qquad \frac{(B \vee C) \vee (B \vee C)}{(B \vee B) \vee (C \vee C)} \qquad \mathsf{m}^{\vee}_{\exists}\,\frac{(\exists x.\, A) \vee (\exists x.\, A)}{\exists x.\, (A \vee A)} \qquad \mathsf{m}^{\vee}_{\forall}\,\frac{(\forall x.\, A) \vee (\forall x.\, A)}{\forall x.\, (A \vee A)}$$

The medial therefore allows us to swap the scope of connectives, bringing together the like parts of the formulae to be contracted, until we reach the atoms. This can be extended to first-order logic using the quantified medials shown in the second two rules above.

However, in first-order logic the atomic contraction $\mathsf{ac}{\downarrow}\,\dfrac{\mathsf{p}\vec{t} \vee \mathsf{p}\vec{t}}{\mathsf{p}\vec{t}}$ is not checkable with constant cost: its complexity depends on the size of the terms in $\vec{t}$. We would like to extend this method inside atomic formulae, giving contraction rules that operate directly on the terms. We can expose the internal structure of terms and atoms using non-commutative, non-associative binary *application* connectives.

▶ **Definition 10** (arities)**.** *The sets of* **compound** *variables, terms and atoms will be stratified into arities, $\mathcal{V}_i$, $\mathcal{T}_i$, and $\mathcal{A}_i$, respectively, for each $i \in N$, with the following recursive grammar:*

$$\mathcal{T}_i ::= \mathcal{V}_i \mid \mathcal{C}_i \mid \mathcal{T}_{i+1} \mathbin{@_T} \mathcal{T}_0 \qquad\qquad \mathcal{A}_i ::= \mathcal{P}_i \mid \overline{\mathcal{P}_i} \mid \mathcal{A}_{i+1} \mathbin{@_i} \mathcal{T}_0$$

Thus, for any $i \in N$, given a compound term $f \in \mathcal{T}_{i+1}$, compound atom $p \in \mathcal{A}_i$, and a term $t \in \mathcal{T}_0$, the **application** *$(f \mathbin{@_T} t)$ is a compound term, and $(p \mathbin{@_i} t)$ is a compound atom, respectively,* **at arity** *i; the application is called* **partial** *iff $i \neq 0$. We reserve the unmodified nouns* term *and* atom*, and the set names $\mathcal{T}$ and $\mathcal{A}$, to refer to $\mathcal{T}_0$ and $\mathcal{A}_0$ respectively. Notationally we assume $@_T$ and $@_i$ to be left-associative and having the highest precedence of all binary operators.*

▶ **Example 11.** If $\mathsf{p} \in \mathcal{P}_2$, $x \in \mathcal{V}_0$, $\mathsf{c} \in \mathcal{C}_0$, and $\mathsf{f} \in \mathcal{C}_1$, then the atom $\mathsf{p}(\mathsf{c}, \mathsf{f}(x))$ is written as $(\mathsf{p} \mathbin{@_1} \mathsf{c}) \mathbin{@_0} (\mathsf{f} \mathbin{@_T} x)$, and $\mathsf{p} \mathbin{@_1} \mathsf{c}$ is a partial atom in $\mathcal{A}_1$.

The purpose of generalising terms and atoms to applications is that we can push contraction inside them, and contract directly on predicates, constants, and variables. For instance, to perform a contraction on the atom in Example 11 amounts to the following derivation:

$$\begin{array}{c} \Big((\mathsf{p} \mathbin{@_1} \mathsf{c}) \mathbin{@_0} (\mathsf{f} \mathbin{@_T} x)\Big) \vee \Big((\mathsf{p} \mathbin{@_1} \mathsf{c}) \mathbin{@_0} (\mathsf{f} \mathbin{@_T} x)\Big) \\ \Vert \\ \left(\mathsf{ac}{\downarrow}\,\dfrac{\mathsf{p} \vee \mathsf{p}}{\mathsf{p}} \mathbin{@_1} \mathsf{tc}{\downarrow}\,\dfrac{\mathsf{c} \sqcup \mathsf{c}}{\mathsf{c}}\right) \mathbin{@_0} \left(\mathsf{tc}{\downarrow}\,\dfrac{\mathsf{f} \sqcup \mathsf{f}}{\mathsf{f}} \mathbin{@_T} \mathsf{tc}{\downarrow}\,\dfrac{x \sqcup x}{x}\right) \end{array}$$

where the only contractions are on truly constant sized objects: the predicates, constants, and variables. Observe that when we push these contractions to constants and variables, we need to introduce a notation for representing these term *duplications*, which we write using $\sqcup$ to be reminiscent of propositional disjunction. For cocontraction we will also need the dual notion of term *de-duplication*, written using $\sqcap$.

▶ **Definition 12** (compound terms)**.** *To terms of every arity, we add the operators $\sqcup$ and $\sqcup$. To support reasoning on these operators, we generalise terms to* **compound terms of arity** *i for every $i \in N$, with the following grammar at every arity.*

$$\mathcal{T}_i ::= \mathcal{V}_i \mid \mathcal{C}_i \mid \mathcal{T}_i \sqcup_i \mathcal{T}_I \mid \mathcal{T}_i \sqcap_i \mathcal{T}_i \mid \frac{\mathcal{T}_i}{\mathcal{T}_i}\,i \mid \mathcal{T}_{i+1} \mathbin{@_T} \mathcal{T}_0$$

We want to avoid nonsense such as the (de-)duplication of terms of different arities, so we annotate the $\sqcup$ and $\sqcap$ connectives with the arities of the terms they are joining. We will generally omit the arity subscripts unless they are essential, and simply assume that compound terms are grammatically (i.e., arity-wise) correct. The set of **nonvertical terms***, $\ddot{\mathcal{T}}_i$, are those terms in $\mathcal{T}_i$ without any occurrence of (vertical) composition by inference.*

Observe also that in our desired contraction derivation above we performed a contraction on p in isolation, which is not a formula but a compound atom of arity 2. Indeed, we have to generalise our notion of derivations to compound derivations to account for such steps.

▶ **Definition 13** (compound atoms). *Like with terms, we stratify* **compound atoms**, *defining for each $i \in N$ the set $\mathcal{A}_i$* **of arity** *i with the following recursive grammar:*

$$\mathcal{A}_i ::= \mathcal{P}_i \mid \mathcal{A}_i \vee_i \mathcal{A}_i \mid \mathcal{A}_i \wedge_i \mathcal{A}_i \mid \frac{\mathcal{A}_i}{\mathcal{A}_i}\,i \mid \mathcal{A}_{i+1} @_i \mathcal{T}_0$$

Like with compound terms, we want to avoid nonsense such as the disjunction of two atoms of different arity, so every connective is annotated with the arity; as before we will generally drop these annotations and assume grammatical correctness. The set of **nonvertical atoms**, *$\ddot{\mathcal{A}}_i$, are those atoms in $\mathcal{A}_i$ without any occurrence of (vertical) composition by inference.*

This more liberal composition of terms and atoms is reminiscent of deep-inference *subatomic systems* for propositional logic [2, 3], extended to predicates. In particular, we see that the necessary rules for decomposing atomic contraction follow the same shape as the medial rule and atomic contractions.

$$\mathsf{m}^{\vee}_{@}\,\frac{(A @_i s) \vee_i (B @_i t)}{(A \vee_{i+1} B) @_i (s \sqcup_0 t)} \qquad \mathsf{m}^{\sqcup}_{@}\,\frac{(f @_T s) \sqcup_i (g @_T t)}{(f \sqcup_{i+1} g) @_T (s \sqcup_0 t)} \qquad \mathsf{pc}{\downarrow}\,\frac{\mathsf{p} \vee_i \mathsf{p}}{\mathsf{p}} \qquad \mathsf{tc}{\downarrow}\,\frac{c \sqcup_i c}{c}$$

▶ **Example 14.** We can now finish the contraction derivation hinted at above:

$$\mathsf{m}^{\vee}_{@}\,\frac{\Big((\mathsf{p} @_1 \mathsf{c}) @_0 (\mathsf{f} @_T x)\Big) \vee_0 \Big((\mathsf{p} @_1 \mathsf{c}) @_0 (\mathsf{f} @_T x)\Big)}{\left(\mathsf{m}^{\vee}_{@}\,\dfrac{(\mathsf{p} @_1 \mathsf{c}) \vee_1 (\mathsf{p} @_1 \mathsf{c})}{\mathsf{ac}{\downarrow}\dfrac{\mathsf{p} \vee_2 \mathsf{p}}{\mathsf{p}} @_1 \mathsf{tc}{\downarrow}\dfrac{\mathsf{c} \sqcup_0 \mathsf{c}}{\mathsf{c}}}\right) @_0 \left(\mathsf{m}^{\sqcup}_{@}\,\dfrac{(\mathsf{f} @_T x) \sqcup_0 (\mathsf{f} @_T x)}{\mathsf{tc}{\downarrow}\dfrac{\mathsf{f} \sqcup_1 \mathsf{f}}{\mathsf{f}} @_T \mathsf{tc}{\downarrow}\dfrac{x \sqcup_0 x}{x}}\right)}$$

3.2 Localising Weakening

The general weakening rule, $\mathsf{w}{\downarrow}\,\dfrac{\bot}{A}$, viewed premise-downwards, creates a formula from nowhere, so checking the correctness of an instance means checking that what is created is indeed a formula (an atomic formula in the case of atomic weakening). Like with contraction, we push the weakening inside atoms so that each check can be done in constant time; we do this by enabling the proof system to construct an atom. Note that this will require us to define weakening the predicates and constants used to build atoms and terms, and therefore we will need to add such units to the definitions of compound atoms, terms, and derivations.

▶ **Definition 15** (syntax with compound units). *For every arity $i \in N$:*

- *We add two* **term units** *$\triangledown_i$ and $\blacktriangle_i$, standing respectively for a deleted and a co-deleted constant. Definition 12 is updated as follows: $\mathcal{T}_i ::= \cdots \mid \triangledown_i \mid \blacktriangle_i$.*
- *We add two* **atomic units** *$\bot_i$ and $\top_i$, standing respectively for a weakened and a co-weakened literal. Definition 13 is updated as follows: $\mathcal{A}_i ::= \cdots \mid \bot_i \mid \top_i$.*
- *We write $\mathcal{W}_i$ for $\mathcal{L}_i \cup \mathcal{C}_i \cup \mathcal{V}_i \cup \{\bot_i, \triangledown_i, \top_i, \blacktriangle_i\}$ and w to denote elements of $\mathcal{W}_i$.*
- *The derivation units $\bot$ and $\top$ are removed from the grammar of derivations (Definition 2); they are just notations for $\bot_0$ and $\top_0$ respectively. We also write $\vee_0$ and $\wedge_0$ instead of $\vee$ and $\wedge$ respectively in derivations.*

$$\mathsf{m}^{\vee}_{@}\,\dfrac{(A \mathbin{@_i} s) \vee_i (B \mathbin{@_i} t)}{(A \vee_{i+1} B) \mathbin{@_i} (s \sqcup_0 t)} \qquad \mathsf{pc}{\downarrow}\,\dfrac{\mathsf{p} \vee_i \mathsf{p}}{\mathsf{p}}\,i \qquad \mathsf{m}^{@}_{\wedge}\,\dfrac{(A \wedge_{i+1} B) \mathbin{@_i} (s \sqcap_0 t)}{(A \mathbin{@_i} s) \wedge_i (B \mathbin{@_i} t)} \qquad \mathsf{tc}{\uparrow}\,\dfrac{\mathsf{p}}{\mathsf{p} \wedge_i \mathsf{p}}\,i$$

$$\mathsf{m}^{\sqcup}_{@}\,\dfrac{(f \mathbin{@_T} s) \sqcup_i (g \mathbin{@_T} t)}{(f \sqcup_{i+1} g) \mathbin{@_T} (s \sqcup_0 t)} \qquad \mathsf{tc}{\downarrow}\,\dfrac{c \sqcup_i c}{c}\,i \qquad \mathsf{m}^{@}_{\sqcap}\,\dfrac{(f \sqcap_{i+1} g) \mathbin{@_T} (s \sqcap_0 t)}{(f \mathbin{@_T} s) \sqcap_i (g \mathbin{@_T} t)} \qquad \mathsf{tc}{\uparrow}\,\dfrac{c}{c \sqcap_i c}\,i$$

$$\bot\triangledown{\downarrow}\,\dfrac{\bot_i}{\bot_{i+1} \mathbin{@_i} \triangledown_0} \qquad \mathsf{pw}{\downarrow}\,\dfrac{\bot_i}{\mathsf{p}}\,i \qquad \top\blacktriangle{\uparrow}\,\dfrac{\top_{i+1} \mathbin{@_i} \blacktriangle_0}{\top_i} \qquad \mathsf{pw}{\uparrow}\,\dfrac{\mathsf{p}}{\top_i}\,i$$

$$\triangledown\triangledown{\downarrow}\,\dfrac{\triangledown_i}{\triangledown_{i+1} \mathbin{@_T} \triangledown_0} \qquad \mathsf{tw}{\downarrow}\,\dfrac{\triangledown_i}{c}\,i \qquad \blacktriangle\blacktriangle{\uparrow}\,\dfrac{\blacktriangle_{i+1} \mathbin{@_T} \blacktriangle_0}{\blacktriangle_i} \qquad \mathsf{tw}{\uparrow}\,\dfrac{c}{\blacktriangle_i}\,i$$

$$\mathsf{m}\,\dfrac{(A \wedge_i B) \vee_i (C \wedge_i D)}{(A \vee_i C) \wedge_i (B \vee_i D)} \qquad \mathsf{m}^{\sqcup}_{\sqcap}\,\dfrac{(q \sqcap_i r) \sqcup_i (s \sqcap_i t)}{(q \sqcup_i s) \sqcap_i (r \sqcup_i t)} \qquad \mathsf{s}^{*}_{@}{\downarrow}\,\dfrac{(A \mathbin{@_i} t) *_i \dagger_i}{(A *_{i+1} \dagger_{i+1}) \mathbin{@_i} t}$$

for $* \in \{\vee, \wedge\}$ and $\dagger \in \{\bot, \top\}$

$$(X *_i Y) \equiv_i (Y *_i X) \qquad X *_i (Y *_i Z) \equiv_i (X *_i Y) *_i Z \qquad \text{for } *_i \in \{\vee_i, \wedge_i, \sqcup_i, \sqcap_i\}$$

$$A \vee_i \bot_i \equiv_i A \wedge_i \top_i \equiv_i A \qquad \top_i \vee_i \top_i \equiv_i \top_i \qquad \bot_i \wedge_i \bot_i \equiv_i \bot_i$$

$$t \sqcup_i \triangledown_i \equiv_i t \sqcap_i \blacktriangle_i \equiv_i t \qquad \blacktriangle_i \sqcup_i \blacktriangle_i \equiv_i \blacktriangle_i \qquad \triangledown_i \sqcap_i \triangledown_i \equiv_i \triangledown_i$$

Figure 2 System SKSql: the rules and equational theory to add to system SKSq (Figure 1). In tc↓, tw↓, tc↑, and tw↑, recall that $c \in \mathcal{V}_i \cup \mathcal{C}_i \cup \{\triangledown_i, \blacktriangle_i\}$. The equational theory is given for both compound terms and atoms for every arity $i \in N$, written $\equiv_i$; in the generating equations, X, Y, Z are assumed to belong to a suitable grammatical category and arity.

The term unit $\triangledown_i$ can be thought of as a *minimal term* of arity i, which can then be used to weaken the components of a term systematically by means of the following four rules; the first show how $@_i$ and $@_T$ are absorbed, and the second two are just the weakening rules on predicates and constants.

$$\bot\triangledown{\downarrow}\,\dfrac{\bot_i}{\bot_{i+1} \mathbin{@_i} \triangledown_0} \qquad \triangledown\triangledown{\downarrow}\,\dfrac{\triangledown_i}{\triangledown_{i+1} \mathbin{@_T} \triangledown_0} \qquad \mathsf{pw}{\downarrow}\,\dfrac{\bot_i}{\mathsf{p}}\,i \qquad \mathsf{tw}{\downarrow}\,\dfrac{\triangledown_i}{c}\,i$$

Observe that for the rules pw↓ and tw↓, the arity of the vertical composition is inherited by that of the compound atomic or term unit. We assume that the inferences are arity-correct, so we could equivalently have deduced the arity of the unit from that of the conclusion, but keeping the arities in the rule is obviously checkable with constant cost.

▶ **Example 16.** The SKSq weakening rule $\mathsf{aw}{\downarrow}\,\dfrac{\bot}{\mathsf{p}(\mathsf{c}, \mathsf{f}(x))}$ has the following localised form:

$$\bot\triangledown{\downarrow}\,\dfrac{\bot_0}{\left(\bot\triangledown{\downarrow}\,\dfrac{\bot_1}{\mathsf{pw}{\downarrow}\,\dfrac{\bot_2}{\mathsf{p}} \mathbin{@_1} \mathsf{tw}{\downarrow}\,\dfrac{\triangledown_0}{\mathsf{c}}}\right) \mathbin{@_0} \left(\triangledown\triangledown{\downarrow}\,\dfrac{\triangledown_0}{\mathsf{tw}{\downarrow}\,\dfrac{\triangledown_1}{\mathsf{f}} \mathbin{@_T} \mathsf{tw}{\downarrow}\,\dfrac{\triangledown_0}{x}}\right)}$$

▶ **Definition 17** (SKSql)**.** *The system* SKSql *consists of the rules of* SKSq *(Figure 1) with the atomic rules* ac↓*,* ac↑*,* aw↓*,* aw↑ *replaced with the rules in Figure 2, and the equational theory of compound terms and atoms added.*

We conclude this section by showing some properties of the interaction between the higher-arity units and term calculus (proofs in Appendix B).

▶ **Proposition 18.** *For any $n \in N$ and terms $t_1, \ldots, t_n$, the compound atom $\bot_n @ t_1 @ \ldots @ t_n$ is equivalent to $\bot_0$.* ◀

▶ **Proposition 19.** *Let A be a $(i+1)$-ary compound atom. Then for any terms s and t, $A @_i s$ implies $A @_i (s \sqcup_i t)$, and $A @_i (s \sqcap_i t)$ implies $A @_i t$.* ◀

4 Localising Instantiation and Vacuousness

The instantiation rule $\mathsf{n}\!\downarrow \dfrac{[t/x]\, A}{\exists x A}$ is doubly non-local because t can be arbitrarily large and x can occur arbitrarily many times in A. We can decompose instantiation into two phases, corresponding to these two sources of non-locality: first t is built incrementally, and then it replaces x in A. The intermediate stages of this incremental treatment of instantiation are represented using *explicit substitution* [1, 3].

We write $[t/x]\, X$ for the actual substitution of a variable x by a term t in in any (compound) term, atom, or formula X, $\langle t/x\rangle\, A$ for the explicit substitution which has not been applied yet, just indicated in the syntax; applying it will result in $[t/x]\, X$.

$$\mathsf{x}\!\downarrow \frac{\exists_{i+1} y \exists_0 z\, \langle y @_T z/x\rangle_i}{\exists_i x} \qquad \mathsf{qn}\!\downarrow \frac{\langle c/x\rangle_i}{\exists_i x}$$

In the first phase, we will build an explicit substitution equivalent to $\langle t/x\rangle$ locally by building the term incrementally, using quantification over higher arity variables. This phase has nothing to do with the target of the substitution X, so we can operate directly inside the binding, transforming it from an existential quantifier to an explicit substitution. The inference rules that we will use to build this compound substitution are shown to the right, which we call *expansion* and *instantiation* respectively (where $c \in \mathcal{C}_i \cup \mathcal{V}_i$ as usual). We implement $\mathsf{x}\!\downarrow$ in such a way that y and z are globally fresh.[4] Thus, in the premise of $\mathsf{x}\!\downarrow$, the modifed $\exists$-prefix, $\exists_{i+1} y \exists_0 z\, \langle y @_T z/x\rangle_i$, acts as a binder for x. Note that since the arity of the variables have increased, both the quantifier and the explicit substitution will need to track the arity.

▶ **Example 20.** Here is the implementation of $\mathsf{n}\!\downarrow$ for the witness term $\mathsf{g}(\mathsf{f}(\mathsf{c}), u)$:

$$\dfrac{\dfrac{\dfrac{\langle \mathsf{g}/y_1\rangle_2}{\exists_2 y_1}\ \dfrac{\dfrac{\langle \mathsf{f}/w_1\rangle_1}{\exists_1 w_1}\ \dfrac{\langle \mathsf{c}/w_2\rangle}{\exists_0 w_2}\ \langle w_1 @ w_2/y_2\rangle_0}{\exists_0 y_2}\ \langle y_1 @_T y_2/y\rangle_1}{\exists_1 y}\ \dfrac{\langle u/z\rangle_0}{\exists_0 z}\ \langle y @_T z/x\rangle_0}{\exists_0 x}\ A$$

where the variables y, z, y_1, y_2, w_1, and w_2 are fresh. The compound substitution

$$\langle \mathsf{g}/y_1\rangle_2\, \langle \mathsf{f}/w_1\rangle_1\, \langle \mathsf{c}/w_2\rangle_0\, \langle w_1 @_T w_2/y_2\rangle_0\, \langle y_1 @_T y_2/y\rangle_1\, \langle u/z\rangle_0\, \langle y @_T z/x\rangle_0$$

is equivalent to $\langle \mathsf{g}(\mathsf{f}(\mathsf{c}), u)/x\rangle$. The tree structure of the term is transferred into the explicit substitutions; $\langle \mathsf{f}/w_1\rangle_1$ and $\langle \mathsf{c}/w_2\rangle_0$ are as independent as the existential quantifiers $\exists_1 w_1$ and $\exists_0 w_2$ in sequence.

▶ **Definition 21** (explicit substitution syntax, updating Definition 15)**.** *For every $i, j \in N$,*

- **Compound terms** *are extended as follows: $\mathcal{T}_i ::= \cdots \mid \langle \mathcal{T}_j/x\rangle_j\, \mathcal{T}_i$.*

---

[4] We assume a form of Barendregt's variable naming convention, where every binder in the entire derivation binds a unique variable. Thus, when we say a variable is "fresh" in an inference rule, we merely mean that the rule introduces its binder and therefore the variable is globally distinct. Appendix E contains a discussion about the costs of creating fresh variables.

- **Compound atoms** *are extended as follows:* $\mathcal{A}_i ::= \cdots \mid \langle \mathcal{T}_j/x\rangle_j\, \mathcal{A}_i$.
- **Prefix derivations** $\mathcal{Q}$ *generalise the quantifiers and have the following grammar:* $\mathcal{Q} ::= \exists_i x \mid \forall_i x \mid \langle t/x\rangle_i \mid \mathcal{Q}\,\mathcal{Q} \mid \dfrac{\mathcal{Q}}{\mathcal{Q}}$. *The* **prefix premise** *and* **prefix conclusion** *maps* $\mathsf{pr}, \mathsf{cn} : \mathcal{Q} \times \mathcal{F} \to \mathcal{F}$ *have:* $\mathsf{pr}(\exists_i x, A) = \mathsf{cn}(\exists_i x, A) = \exists_i x.\, A$, $\mathsf{pr}(\forall_i x, A) = \mathsf{cn}(\forall_i x, A) = \forall_i x.\, A$, $\mathsf{pr}(\langle t/x\rangle_i, A) = \mathsf{cn}(\langle t/x\rangle_i, A) = \langle t/x\rangle_i\, A$, $\mathsf{pr}(Q\,Q', A) = \mathsf{pr}(Q, \mathsf{pr}(Q', A))$, $\mathsf{cn}(Q\,Q', A) = \mathsf{cn}(Q, \mathsf{cn}(Q', A))$, $\mathsf{pr}\left(\dfrac{Q}{Q'}, A\right) = \mathsf{pr}(Q, A)$, *and* $\mathsf{cn}\left(\dfrac{Q'}{Q}, A\right) = \mathsf{cn}(Q, A)$.
- **Compound derivations** *are modified by removing the quantified derivations* $\exists_i x.\, \mathcal{D}$ *and* $\forall_i x.\, \mathcal{D}$, *and adding* $\mathcal{D} ::= \cdots \mid \mathcal{Q}\,\mathcal{D}$. *The prefix and conclusion maps for derivations are updated as follows* $\mathsf{pr}(Q\ \Phi) = \mathsf{pr}(Q, \mathsf{pr}(\Phi))$ *and* $\mathsf{cn}(Q\ \Phi) = \mathsf{cn}(Q, \mathsf{cn}(\Phi))$.

For the second phase of instantiation, we successively push the component substitutions the body, with inference rules like on the right for various operators $*$. Finally, once an explicit substitution reaches a leaf node (constant, predicate, or variable), it disappears, either by being applied successfully or vacuously. By construction, each substitution term created in the first phase is small, only a single element or a single elementary application, so we don't need to duplicate arbitrarily large terms. Explicit substitutions neither create nor destroy information, so we might expect that this can be done via invertible inference rules.

$$\frac{\langle t/x\rangle_i\, B * \langle t/x\rangle_i\, C}{\langle t/x\rangle_i (B * C)}$$

However, there is a complication due to the interaction with the compound term units and connectives introduced in Section 3.1. For example, consider the following derivations:[5]

$$\dfrac{\dfrac{\mathsf{p} \mathbin{@} \mathsf{c}}{\mathsf{p} \mathbin{@} (\mathsf{c} \sqcup \mathsf{d})}\,\text{Prop.19} \;\wedge_0\; \dfrac{\bar{\mathsf{p}} \mathbin{@} \mathsf{d}}{\bar{\mathsf{p}} \mathbin{@} (\mathsf{c} \sqcup \mathsf{d})}\,\text{Prop.19}}{\dfrac{\langle \mathsf{c} \sqcup_0 \mathsf{d}/x\rangle_0 \ \mathsf{ai}{\uparrow}\dfrac{(\mathsf{p} \mathbin{@} x) \wedge_0 (\bar{\mathsf{p}} \mathbin{@} x)}{\bot_0}}{\bot_0}} \qquad \text{and} \qquad \dfrac{\mathsf{tw}{\uparrow}\dfrac{\mathsf{c}}{\blacktriangle_0} \neq \mathsf{tw}{\uparrow}\dfrac{\mathsf{d}}{\blacktriangle_0}}{\dfrac{\langle \blacktriangle_0/x\rangle_0 \dfrac{x \neq x}{\bot_0}}{\bot_0}}$$

We should not be able to derive a contradiction from either $\mathsf{p}(\mathsf{c}) \wedge \bar{\mathsf{p}}(\mathsf{d})$ or $\mathsf{c} \neq \mathsf{d}$ (dually, we should not be able to prove $\mathsf{p}(\mathsf{c}) \vee \bar{\mathsf{p}}(\mathsf{d})$ or $\mathsf{c} = \mathsf{d}$). The mistake, marked by the squiggly lines, is in naïvely allowing $\mathsf{c} \sqcup_0 \mathsf{d}$ and $\blacktriangle_0$ to be duplicated as they distribute over $\wedge$ and @[6]. Composition by explicit substitution does not interact straight-forwardly with duality [3, 14] , and by introducing units $\triangledown_i$ and $\blacktriangle_i$ and connectives $\sqcup_i$ and $\sqcap_i$, we have exposed terms to duality. The negation of $\mathsf{p} \mathbin{@_1} (\mathsf{c} \sqcup_0 \mathsf{d})$ is not $\bar{\mathsf{p}} \mathbin{@_0} (\mathsf{c} \sqcup_0 \mathsf{d})$ but rather $\bar{\mathsf{p}} \mathbin{@_0} (\mathsf{c} \sqcup_0 \mathsf{d})$, and explicit substitutions shouldn't allow us to cheat the cut rule in this way. We can enforce correctness by restricting true duplication upwards over $\wedge$ and @ to terms that cannot be transformed in this way; that is, ordinary terms and $\triangledown_i$, which cannot create new material upwards.

▶ **Definition 22.** *A term* $t \in \mathcal{T}_i$ *is* **elementary** *if it is in* $\mathcal{C}_i \cup \mathcal{V}_i$ *or* $(\mathcal{C}_{i+1} \cup \mathcal{V}_{i+1}) \mathbin{@_T} (\mathcal{C}_0 \cup \mathcal{V}_0)$. *We denote the set of elementary terms of arity* i *by* $\mathcal{E}_i$.

Elementary terms are both ordinary and small. Given an explicit substitution $\langle t/x\rangle_i$, we can check in constant time whether t is elementary, and we restrict duplicating rules to the cases where t is either elementary or $\triangledown_i$.[7]

---

[5] In the derivation on the right, we imagine that we have a distinguished predicate $\neq$ for term inequality, simply because this better highlights the problem.

[6] We understand the structure of $x \neq y$ to be $\neq \mathbin{@} x \mathbin{@} y$, so the connective really is @.

[7] See Example 31 for how the problematic derivations are fixed.

$$\mathsf{x}{\downarrow}\,\frac{\exists_{i+1} y \exists_0 z\, \langle y \mathbin{@_T} z/x\rangle_i}{\exists_i x} \qquad \mathsf{qn}{\downarrow}\,\frac{\langle c/x\rangle_i}{\exists_i x} \qquad \text{where } c \in \mathcal{C}_i \cup \mathcal{V}_i \text{ and } y, z \text{ are fresh}$$

$$\sigma{\downarrow}\,\frac{t}{\langle t/x\rangle_i\, x} \qquad \mathsf{v}\sigma{\downarrow}\,\frac{w}{\langle t/x\rangle_i\, w} \qquad \sigma\sigma{\downarrow}\,\frac{\big\langle \langle e/x\rangle_i\, s/y\big\rangle_k\, \langle e/x\rangle_i\, X}{\langle e/x\rangle_i\, \langle s/y\rangle_k\, X} \qquad \begin{array}{r}\text{for } w \in \mathcal{W}_i \setminus \{x\}\,,\\ P \in \mathcal{A}_0, e \in \mathcal{E}_i,\\ X \in \mathcal{T}_j \cup \mathcal{A}_j\end{array}$$

$$\left.\begin{array}{ll}\sigma{@_T}{\downarrow}\,\dfrac{(\langle e/x\rangle_i\, f) \mathbin{@_T} (\langle e/x\rangle_i\, s)}{\langle e/x\rangle_i (f \mathbin{@_T} s)} & \sigma{@}{\downarrow}\,\dfrac{(\langle e/x\rangle_i\, A) \mathbin{@_j} (\langle e/x\rangle_i\, s)}{\langle e/x\rangle_i (A \mathbin{@_j} s)} \\[2ex] \sigma{\sqcap}{\downarrow}\,\dfrac{(\langle e/x\rangle_i\, s) \sqcap_j (\langle e/x\rangle_i\, t)}{\langle e/x\rangle_i (s \sqcap_j t)} & \sigma{\wedge}{\downarrow}\,\dfrac{(\langle e/x\rangle_i\, A) \wedge_j (\langle e/x\rangle_i\, B)}{\langle e/x\rangle_i (A \wedge_j B)}\end{array}\right\}\ \text{for } e \in \mathcal{E}_i$$

$$\mathsf{m}^{\sqcup}_{\sigma}\,\frac{(\langle s/x\rangle_i\, q) \sqcup_j (\langle t/x\rangle_i\, r)}{\langle s \sqcup_i t/x\rangle_i (q \sqcup_j r)} \qquad \mathsf{m}^{\vee}_{\sigma}\,\frac{(\langle s/x\rangle_i\, A) \vee_j (\langle t/x\rangle_i\, B)}{\langle s \sqcup_i t/x\rangle_i (A \vee_j B)} \qquad \sigma\mathsf{Q}{\downarrow}\,\frac{\mathsf{Q}_j y.\, \langle t/x\rangle_i\, A}{\langle t/x\rangle_i\, \mathsf{Q}_j y.\, A} \qquad \text{for } \mathsf{Q} \in \{\exists, \forall\}$$

$$\left.\begin{array}{lll}\mathsf{qv}{\downarrow}\,\dfrac{\exists_i x}{\langle \blacktriangle_i/x\rangle_i} & \blacktriangle\mathsf{v}{\downarrow}\,\dfrac{\langle \blacktriangle_i/x\rangle_i\, w}{w} & \blacktriangle{*}{\downarrow}\,\dfrac{\langle \blacktriangle_j/x\rangle_j (X *_i Y)}{\langle \blacktriangle_j/x\rangle_j\, X *_i \langle \blacktriangle_j/x\rangle_j\, Y} \\[2ex] \blacktriangle\sigma{\downarrow}\,\dfrac{\langle \blacktriangle_i/x\rangle_i\, \langle s/y\rangle_k\, X}{\big\langle \langle \blacktriangle_i/x\rangle_i\, s/y\big\rangle_k\, \langle \blacktriangle_i/x\rangle_i\, X} & & \blacktriangle\mathsf{Q}{\downarrow}\,\dfrac{\langle \blacktriangle_i/x\rangle_i\, \mathsf{Q}_j y.\, A}{\mathsf{Q}_j y.\, \langle \blacktriangle_i/x\rangle_i\, A}\end{array}\right\}\ \begin{array}{l}\text{for } * \in \{\vee_i, \wedge_i, @_i, @_T, \sqcup_i, \sqcap_i\}\\ \text{suitable X, Y, and } \mathsf{Q} \in \{\exists, \forall\}\end{array}$$

Figure 3 System SKSqlx: the rules to add to system SKSql (Figure 2). Only the down-rules are shown; the symmetric rules are obtained by dualising.

We can also apply a substitution $\langle s \sqcup_i t/x_i\rangle$ by means of the medial rule shown to the right, which allows us to extend contraction to formulae with explicit substitutions. $\mathsf{m}^{\vee}_{\sigma}\,\dfrac{(\langle s/x\rangle_i\, A) \vee_0 (\langle t/x\rangle_i\, A)}{\langle s \sqcup_i t/x\rangle_i (A \vee_0 B)}$ In the case of the example on the left, contracting $(\mathsf{p} \mathbin{@_0} x) \wedge_0 (\bar{\mathsf{p}} \mathbin{@_0} x)$ and applying this medial would result in a sound derivation with premise $(\mathsf{p}(\mathsf{c}) \wedge \bar{\mathsf{p}}(\mathsf{c})) \vee (\mathsf{p}(\mathsf{d}) \wedge \bar{\mathsf{p}}(\mathsf{d}))$.

In the case of pushing an explicit substitution into the operands of $@_i$ or $@_T$, the nature of the substitution changes: while inference rules may still be applied on the terms inside the substitution, the substitution itself cannot transform into a quantifier, since $\exists x.\, \Phi$ is a derivation and not a (compound) atom, i.e., $(\exists x.\, \Phi) \mathbin{@} t$ is outside the grammar. To achieve this, we distinguish explicit substitutions on subatomic particles in the form $\langle t/x\rangle_i$ which cannot turn into quantifiers.

▶ **Example 23.** The substitution $\langle \mathsf{f}(y)/x\rangle$ applied to $\mathsf{p}(x) \vee_0 (\forall_0 y.\, \mathsf{q})$ moves as follows:

$$\mathsf{m}^{\vee_0}_{\sigma}\,\frac{@_0\sigma{\downarrow}\,\dfrac{\sigma{\downarrow}\,\dfrac{\mathsf{p}}{\langle \mathsf{f} \mathbin{@_T} y/x\rangle_0\, \mathsf{p}} \mathbin{@_0} \sigma{\downarrow}\,\dfrac{\mathsf{f} \mathbin{@_T} y}{\langle \mathsf{f} \mathbin{@_T} y/x\rangle_0\, x}}{\langle \mathsf{f} \mathbin{@_T} y/x\rangle_0 (\mathsf{p} \mathbin{@_0} x)} \vee_0 \forall\sigma{\downarrow}\,\dfrac{\forall_0 y\, \sigma{\downarrow}\,\dfrac{\mathsf{q}}{\langle \mathsf{f} \mathbin{@_T} y/x\rangle\, \mathsf{q}}}{\langle \mathsf{f} \mathbin{@_T} y/x\rangle_0 (\forall_0 y.\, \mathsf{q})}}{\left\langle \mathsf{m}^{\sqcup}_{@}\,\dfrac{(\mathsf{f} \mathbin{@_T} y) \sqcup_0 (\mathsf{f} \mathbin{@_T} y)}{\mathsf{tc}{\downarrow}\,\dfrac{\mathsf{f} \sqcup_1 \mathsf{f}}{\mathsf{f}} \mathbin{@_T} \mathsf{tc}{\downarrow}\,\dfrac{y \sqcup_0 y}{y}} \middle/ x \right\rangle_0 ((\mathsf{p} \mathbin{@_0} x) \vee_0 (\forall_0 y.\, \mathsf{q}))}$$

Vacuous quantification $\mathsf{v}{\downarrow}\,\dfrac{\exists x.\, A}{A}$ (where x does not occur free in A) is not local because it necessitates checking for x in A. By using explicit substitutions, we are able to simulate it locally. If we can apply a substitution $\langle t/x\rangle$ to a formula A, and A is not changed by this, then x must not appear free in A. In fact, it is enough to test with the compound unit $\blacktriangle_i$ instead of t, since it can be co-weakened to t using $\mathsf{tw}{\uparrow}$ if needed. Reading upwards from the conclusion, the leaves in A of the form $w \in \bigcup_j \mathcal{W}_j \setminus \{x\}$ are changed to $\langle \blacktriangle_i/x\rangle_i\, w$,

$$
\mathsf{i}^{\leftrightharpoons}_{\vee}{\downarrow}\,\frac{A \leftrightharpoons_0 B}{A \vee_0 B}
\qquad
\mathsf{pr}{\downarrow}\,\frac{\top_0}{\mathsf{p} \leftrightharpoons_i \bar{\mathsf{p}}}
\qquad
\mathsf{tr}{\downarrow}\,\frac{\top_0}{c \doteq_i c}
\qquad
\mathsf{ur}{\downarrow}\,\frac{\top_0}{\triangledown_i \doteq_i \blacktriangle_i}
$$

$$
\mathsf{ti}^{@}{\downarrow}\,\frac{(f \doteq_{i+1} g) \wedge_0 (s \doteq_0 t)}{(f \mathbin{@_T} s) \doteq_i (g \mathbin{@_T} t)}
\qquad
\mathsf{ti}^{\sqcup}{\downarrow}\,\frac{(q \doteq_i s) \wedge_0 (r \doteq_i t)}{(q \sqcap_i r) \doteq_i (s \sqcup_i t)}
\qquad
\mathsf{ti}^{\sigma}{\downarrow}\,\frac{(q \doteq_j r) \wedge_0 (s \doteq_i t)}{\langle q/x\rangle_j\, s \doteq_i \langle r/x\rangle_j\, t}
$$

$$
\mathsf{pi}^{@}{\downarrow}\,\frac{(A \leftrightharpoons_{i+1} B) \wedge_0 (s \doteq_0 t)}{(A \mathbin{@_T} s) \leftrightharpoons_i (B \mathbin{@_T} t)}
\qquad
\mathsf{pi}^{\vee}{\downarrow}\,\frac{(A \leftrightharpoons_i B) \wedge_0 (C \leftrightharpoons_i D)}{(A \wedge_i C) \leftrightharpoons_i (B \vee_i D)}
\qquad
\mathsf{pi}^{\sigma}{\downarrow}\,\frac{(q \doteq_j r) \wedge_0 (A \leftrightharpoons_i B)}{\langle q/x\rangle_j\, A \leftrightharpoons_i \langle r/x\rangle_j\, B}
$$

$$
\mathsf{w}^{\doteq}{\downarrow}\,\frac{\bot_0}{\triangledown_i \doteq_i \triangledown_i}
\qquad
\mathsf{w}^{\not\doteq}{\downarrow}\,\frac{\bot_0}{\triangledown_i \not\doteq_i \triangledown_i}
\qquad
\mathsf{c}^{\doteq}{\downarrow}\,\frac{(q \doteq_i r) \vee_0 (s \doteq_i t)}{(q \sqcup_i s) \doteq_i (r \sqcup_i t)}
\qquad
\mathsf{c}^{\not\doteq}{\downarrow}\,\frac{(q \not\doteq_i r) \vee_0 (s \not\doteq_i t)}{(q \sqcup_i s) \not\doteq_i (r \sqcup_i t)}
$$

$$
\mathsf{w}^{\leftrightharpoons}{\downarrow}\,\frac{\bot_0}{\bot_i \leftrightharpoons_i \bot_i}
\qquad
\mathsf{w}^{\not\leftrightharpoons}{\downarrow}\,\frac{\bot_0}{\bot_i \not\leftrightharpoons_i \bot_i}
\qquad
\mathsf{c}^{\leftrightharpoons}{\downarrow}\,\frac{(A \leftrightharpoons_i B) \vee_0 (C \leftrightharpoons_i D)}{(A \vee_i C) \leftrightharpoons_i (B \vee_i D)}
\qquad
\mathsf{c}^{\not\leftrightharpoons}{\downarrow}\,\frac{(A \not\leftrightharpoons_i B) \vee_0 (C \not\leftrightharpoons_i D)}{(A \vee_i C) \not\leftrightharpoons_i (B \vee_i D)}
$$

$$
\mathsf{i}^{\doteq}{\downarrow}\,\frac{(q \doteq_i r) \wedge_0 (s \doteq_i t)}{(q \doteq_i s) \vee_0 (r \not\doteq_i t)}
\qquad
\mathsf{i}^{\leftrightharpoons}{\downarrow}\,\frac{(A \leftrightharpoons_i B) \wedge_0 (C \leftrightharpoons_i D)}{(A \leftrightharpoons_i C) \vee_0 (B \not\leftrightharpoons_i D)}
$$

Figure 4 System SKSqt: the rules to add to system SKSqlx (Figure 3). Only the down-rules are shown; the symmetric rules are obtained by dualising.

then these particular substitutions are repeatedly merged until they successfully reach A, at which point it must be that x was vacuous in A so the substitution can be changed into an $\exists$ using the rule $\mathsf{qv}{\downarrow}\,\dfrac{\exists_i x}{\langle \blacktriangle_i/x\rangle_i}$ that is applied entirely within the existential prefix. We want the inference rule to be local, so it must be sound even if x does appear free in A. In this case, the explicit substitution will act like a coweakening, covering every x by $\blacktriangle_i$ to obtain $[\blacktriangle_i/x]_i\, A$. No matter which term witnesses the existential $\exists_i x.\, A$, we would be able to coweaken on it, so $\mathsf{qv}{\downarrow}$ is like an anonymous coweakening.

▶ **Definition 24** (system SKSqlx). *The system* SKSqlx *is obtained from* SKSql *(Definition 17) by adding the rules and equational theory in Figure 3.*

5 Localising Identity with an Equality Predicate

The identity rule $\mathsf{ai}{\downarrow}\,\dfrac{\top}{\mathsf{p}(t_1,\ldots,t_n) \vee \bar{\mathsf{p}}(t_1,\ldots,t_n)}$ is not local because the terms can be arbitrarily large and must be compared. Suppose, given two atomic formulae, $A = \mathsf{p}(t_1,\ldots,t_m)$ and $B = \mathsf{q}(s_1,\ldots,s_n)$, we want to determine whether $\mathsf{ai}{\downarrow}\,\dfrac{\top}{A \vee B}$ is a valid instance of the identity rule. We need to determine whether $B = \overline{A}$, which we can do by checking if $\mathsf{q} = \bar{\mathsf{p}}$ (in which case it follows that $m = n$) and that for all $i \in 1..n$, $t_i = s_i$. Guided by this intuition, we do not decompose the atomic identity rule by pushing it inside the atoms, as we do for contraction and weakening, but rather by exposing a conjunction of all the things to check. Likewise, equality of two ordinary terms can be decomposed by means of a linear rule $\dfrac{(f = s) \wedge_0 (g = t)}{(f \mathbin{@_T} s) = (g \mathbin{@_T} t)}$ and a local reflexivity rule $\dfrac{\top_0}{c = c}$ for $c \in \mathcal{C}_i \cup \mathcal{V}_i$.

However, we must again be careful because compound terms might contain term units $\triangledown_i$, $\blacktriangle_i$ and term connectives $\sqcup_i$, $\sqcap_i$. For example, we don't want to be able to use $\triangledown_0 = \triangledown_0$ to prove $(\mathsf{p} \mathbin{@_0} \triangledown_0) \vee_0 (\bar{\mathsf{p}} \mathbin{@_0} \triangledown_0)$, , and then apply weakening to prove $\mathsf{p}(s) \vee \bar{\mathsf{p}}(t)$ for any terms

s and t. Indeed, the negation of $\mathsf{p} \mathbin{@_0} \triangledown_0$ is $\bar{\mathsf{p}} \mathbin{@_0} \blacktriangle_0$. So our reflexivity rule shouldn't be *"are these two (partial) terms equal?"* but rather *"are these (partial) terms one the dual of the other?"* These relations coincide for all ordinary terms and diverge for any terms containing compound units and connectives. We denote the latter relation by $\doteqdot_i$, where $i \geq 0$ is the arity of the partial terms being compared, and it is governed by the inference rules $\mathsf{tr}\!\downarrow$, $\mathsf{ti}^{@}\!\downarrow$, $\mathsf{ti}^{\sqcup}\!\downarrow$ and $\mathsf{ti}^{\sigma}\!\downarrow$ in Figure 3.

We also note that $\doteqdot_i$ is an intensional equality predicate: its arguments are not terms (except when $i = 0$) but higher arity partial terms. An extensional reading of $\mathsf{f} \doteqdot_2 \mathsf{g}$ could be $\forall_0 x \forall_0 y((\mathsf{f} \mathbin{@_T} x) \mathbin{@_T} y) \doteqdot_0 ((\mathsf{g} \mathbin{@_T} x) \mathbin{@_T} y)$, but this stops us from directly comparing f and g. We adapt the structural rules of contraction, weakening and identity to this quirk as shown in the bottom three lines of Figure 4, noting that the inference rules such as $\mathsf{tw}\!\downarrow \dfrac{\triangledown_i}{c_i}$ and $\triangledown\triangledown\!\downarrow \dfrac{\triangledown_i}{\triangledown_{i+1} \mathbin{@_T} \triangledown_0}$, shown in Figure 3, can still be applied to partial terms in the scope of $\doteqdot_i$ so that weakening can be applied. We further note that coweakening inside terms allows us to prove formulae like $(\mathsf{f} \mathbin{@_T} \blacktriangle_0) \doteqdot_1 (\blacktriangle_2 \mathbin{@_T} \mathsf{c})$, and by Proposition 19, we can prove formulae like $(f_2 \sqcup_2 g_2) \doteqdot_2 f_2$.

The other part of the decomposition of the identity rule concerns comparing predicates. After we have separated off the terms, we are left with two predicates and we want an inference rule that will check whether one is the negation of the other. We introduce another new connective, $\fallingdotseq_i$, for $i \geq 0$, and inference rules $\mathsf{i}^{\fallingdotseq}_{\triangledown}\!\downarrow$, $\mathsf{pr}\!\downarrow$, $\mathsf{pi}^{@}\!\downarrow$, $\mathsf{pi}^{\vee}\!\downarrow$ and $\mathsf{pi}^{\sigma}\!\downarrow$ given in Figure 4 that allow us to decompose compound predicates and compare them. The inference rule $\mathsf{i}^{\fallingdotseq}_{\triangledown}\!\downarrow$ bridges the gap between normal formulae and $\mathsf{pr}\!\downarrow$ which concerns $\fallingdotseq_i$. This equality has an intensional reading similarly to that of $\doteqdot_i$; in particular, it compares (compound) atoms for syntactic equality. We supplement the proof system with structural rules designed for $\fallingdotseq_i$, shown in Figure 4.

▶ **Definition 25** (equalities, updating Definition 21). *For every $i \in N$:*

- **Compound atoms**: $\mathcal{A}_i ::= \cdots \mid \mathcal{T}_i \doteqdot_i \mathcal{T}_i \mid \mathcal{T}_i \not\doteqdot_i \mathcal{T}_i \mid \mathcal{A}_i \fallingdotseq_i \mathcal{A}_i \mid \mathcal{A}_i \not\fallingdotseq_i \mathcal{A}_i$.
- **Duality**: *we extend the involutive duality maps to (composition-by-inference-free) non-vertical terms and atoms generated by the equalities:*

$$\bar{\bar{t}} = t \qquad \overline{c_i} = c_i \ \big(\text{for } c \in \mathcal{C}_i \cup \mathcal{V}_i\big) \qquad \overline{\triangledown_i} = \blacktriangle_i \qquad \overline{s \sqcup_i t} = \bar{s} \sqcap_i \bar{t} \qquad \overline{s \mathbin{@_T} q} = \bar{s} \mathbin{@_T} \bar{q}$$

$$\bar{\bar{A}} = A \qquad \overline{\bot_i} = \top_i \qquad \overline{A \vee_i B} = \bar{A} \wedge_i \bar{B} \qquad \overline{s \doteqdot_i t} = \bar{s} \not\doteqdot_i \bar{t} \qquad \overline{A \fallingdotseq_i B} = \bar{A} \not\fallingdotseq_i \bar{B}$$

$$\overline{\langle s/x \rangle_i\, t} = \langle \bar{s}/x \rangle_i\, \bar{t} \qquad \overline{\langle s/x \rangle_i\, A} = \langle \bar{s}/x \rangle_i\, \bar{B}$$

▶ **Definition 26** (system SKSqt). *The system* SKSqt *is derived from* SKSqlx *(Definition 24) by removing the atomic identity rules* $\mathsf{ai}\!\downarrow$ *and* $\mathsf{ai}\!\uparrow$ *and adding the rules in Figure 4.*

6 Meta-Theory

The first main meta-theorem about SKSqt is is completeness with respect to SKSq.

▶ **Theorem 27** (completeness). *For any $A, B \in \mathcal{F}$, if* $\begin{array}{c} A \\ \| \mathsf{SKSq} \\ B \end{array}$*, then* $\begin{array}{c} A \\ \| \mathsf{SKSqt} \\ B \end{array}$.

Proof Sketch. Each of the atomic rules $\mathsf{ai}\!\downarrow$, $\mathsf{ai}\!\uparrow$, $\mathsf{ac}\!\downarrow$, $\mathsf{ac}\!\uparrow$, $\mathsf{aw}\!\downarrow$, $\mathsf{aw}\!\uparrow$, $\mathsf{av}\!\downarrow$ and $\mathsf{av}\!\uparrow$, and the general instantiation rules $\mathsf{n}\!\downarrow$ and $\mathsf{n}\!\uparrow$, of SKSq will be shown to be derivable in SKSqt. The proof involves building the derivations below for every $i \in N$ by induction:

$$
\underset{\text{(a)}}{\begin{array}{c} t \sqcup_i t \\ \| \,\mathsf{KSqt} \\ t \end{array}}
\quad
\underset{\text{(b)}}{\begin{array}{c} A \vee_i A \\ \| \,\mathsf{KSqt} \\ A \end{array}}
\quad
\underset{\text{(c)}}{\begin{array}{c} \triangledown_i \\ \| \,\mathsf{KSqt} \\ t \end{array}}
\quad
\underset{\text{(d)}}{\begin{array}{c} \bot_i \\ \| \,\mathsf{KSqt} \\ A \end{array}}
\quad
\underset{\text{(e)}}{\begin{array}{c} \sigma^t_{x,i} \\ \| \,\mathsf{KSqt} \\ \exists_i x \end{array}}
\quad
\underset{\text{(f)}}{\begin{array}{c} [s/x]_i\, X \\ \| \,\mathsf{KSqt} \\ \langle s/x\rangle_i\, X \end{array}}
\quad
\underset{\text{(g)}}{\begin{array}{c} \langle \blacktriangle_i/x\rangle_i\, X \\ \| \,\mathsf{KSqt} \\ [\blacktriangle_i/x]_i\, X \end{array}}
\quad
\underset{\text{(h)}}{\begin{array}{c} \top_0 \\ \| \,\mathsf{KSqt} \\ t \mathrel{\mathring{=}}_i \bar{t} \end{array}}
\quad
\underset{\text{(i)}}{\begin{array}{c} \top_0 \\ \| \,\mathsf{KSqt} \\ A \vee_i \overline{A} \end{array}}
$$

Derivations (a) and (b) are used to show ac↓ is admissible, and their duals show that ac↑ is admissible, in SKSqt; (c) and (d) similarly show aw↓ and aw↑; (e) and (f) show n↓ and n↑; (g) establishes av↓ and av↑, where $X \in \mathcal{A}_i \cup \mathcal{F}$, and where $\sigma^t_{x,i}$ is a sequence of elementary explicit substitutions that amount to $\langle t/x\rangle_i$; and (h) and (i) are used to derive ai↓ and ai↑. The details of this sketched proof are in Appendix C. ◀

A more interesting theorem is conservativity: no additional theorems result from the additional structure in SKSqt. We consider it to be a strength of SKSqt that rules can be spread through one another; for example, we can delay firing instantiation instead of *guessing* a term to instantiate. This merging of rules makes it surprising that conservativity nevertheless holds for this system.

▶ **Theorem 28** (conservativity). *Let $A, B \in \mathcal{F}$ be given as in Definition 2, i.e., containing no occurrences of $\{\triangledown, \blacktriangle, \sqcup, \sqcap\}$, nor occurrences of $\{\exists_i, \forall_i, \vee_i, \wedge_i\}$ at arities $i > 0$, nor partially applied compound terms or atoms. If* $\begin{array}{c} A \\ \| \,\mathsf{SKSqt} \\ B \end{array}$ *then* $\begin{array}{c} A \\ \| \,\mathsf{SKSq} \\ B \end{array}$.

Proof Sketch. First we show that SKSqt derivations are conservative over a variant of SKSqlx with ai↓ and ai↑ replaced with similar rules for the atoms of SKSqt. Next we show that this system is conservative over a system with that removes $\sqcup_i, \sqcap_i$ for $i \geq 0$ and $\vee_i, \wedge_i$ for $i > 0$ and replaces the local contraction rules with atomic contraction. Next we show that this system is conservative of a system with $\triangledown_i, \blacktriangle_i$ for $i \geq 0$ and $\bot_i, \top_i$ for $i > 0$ removed and local weakening and local vacuousness replaced by the atomic versions. Finally we show this system is conservative over SKSq. The details of the proof systems and the steps of the conservativity result can be found in Appendix D. ◀

7 Conclusion

We have given a fully local symmetric proof system SKSqt in open deduction for classical first-order logic that is complete and conservative with respect to a non-local system SKSq. The key innovations are: to allow vertical composition by inference in (partially applied) terms, atoms, and quantifier prefixes; to add new operators to denote (co)contraction and (co)weakening within terms and atoms, which are dualised by negation; to use explicit substitutions over higher-arity variables to incrementally build and process instantiations; and to localise (co)identity with new predicates for equality.

An obvious next step is to build a semantics of SKSqt in its own right. A denotational semantics for compound terms and atoms would also establish the global consistency of arbitrary SKSqt derivations, not just those that have ordinary formulas at its endpoints. Moreover, it would be important to develop an internal meta-theory (in particular the up-fragment elimination) for SKSqt. One can certainly adapt the roundabout argument of [13] for the non-local system SKSq, but there is still no fully internal cut-elimination procedure for even SKSq despite being an open problem for 20 years. Full locality might allow us to define generalisations of *atomic flows* from the propositional case, and define abstract normalisation procedures on these flows [11]. Explicit substitutions have been used as a mechanism for proof compression [5]. Along with the other compression mechanisms which are novel to deep-inference, keeping explicit substitutions but forgetting the need for locality could lead to new lower bounds for proofs. Incremental instantiations using higher-arity variables allows for derivations that would not be possible directly in SKSq; for instance, $\begin{array}{c}\exists y.\,\mathsf{p}(\mathsf{f}(y))\\ \Vert\,\mathsf{KSqt}\\ \exists x.\,\mathsf{p}(x)\end{array}$.

References

1 Martín Abadi, Luca Cardelli, Pierre-Louis Curien, and Jean-Jacques Lévy. Explicit substitutions. *Journal of Functional Programming*, 1(4):375–416, October 1991.

2 Andrea Aler Tubella. *A study of normalisation through subatomic logic*. PhD thesis, University of Bath, 2017.

3 Victoria Barrett. *A strictly linear proof system for propositional classical logic*. PhD thesis, University of Bath, 2024.

4 Victoria Barrett, Alessio Guglielmi, and Benjamin Ralph. A strictly linear subatomic proof system. In Jörg Endrullis and Sylvain Schmitz, editors, *EACSL Annual Conference on Computer Science Logic (CSL)*, volume 326 of *LIPIcs*, pages 39:1–39:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. `doi:10.4230/LIPICS.CSL.2025.39`.

5 Victoria Barrett, Alessio Guglielmi, Benjamin Ralph, and Lutz Straßburger. Proof compression via subatomic logic and guarded substitutions. In *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 183–195. IEEE, 2025. `doi:10.1109/LICS65433.2025.00021`.

6 Kai Brünnler. *Deep inference and symmetry in classical proofs*. Logos Verlag Berlin, 2004.

7 Paola Bruscoli and Alessio Guglielmi. On the proof complexity of deep inference. *ACM Trans. on Computational Logic*, 10(2), 2009.

8 Kaustuv Chaudhuri and Nicolas Guenot. Equality and fixpoints in the calculus of structures. In Tom Henzinger and Dale Miller, editors, *Proceedings of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2014. URL: `https://chaudhuri.info/papers/csllics14eqfix.pdf`, `doi:10.1145/2603088.2603140`.

9 Alessio Guglielmi. A system of interaction and structure. *ACM Trans. on Computational Logic*, 8(1):1–64, January 2007. `doi:10.1145/1182613.1182614`.

10 Alessio Guglielmi and Tom Gundersen. Normalisation control in deep inference via atomic flows. *Logical Methods in Computer Science*, 4(1), 2008.

11 Alessio Guglielmi, Tom Gundersen, and Lutz Straßburger. Breaking paths in atomic flows for classical logic. In Jean-Pierre Jouannaud, editor, *25th Symp. on Logic in Computer Science*, pages 284–293, Edinburgh, United Kingdom, July 2010. URL: `https://www.lix.polytechnique.fr/~lutz/papers/AFII.pdf`, `doi:10.1109/LICS.2010.12`.

12 Tom Gundersen, Willem Heijltjes, and Michel Parigot. Atomic lambda calculus: A typed lambda-calculus with explicit sharing. In *28th Symp. on Logic in Computer Science*, pages 311–320. IEEE Computer Society, 2013. URL: `https://dl.acm.org/citation.cfm?id=2591370`, `doi:10.5555/2591370.2591373`.

13 Benjamin Ralph. *Modular normalisation of classical proofs*. PhD thesis, University of Bath, 2019.

14 Alessio Santamaria. *Towards a Godement calculus for dinatural transformations*. PhD thesis, University of Bath, 2019.

15 David Sherratt, Willem Heijltjes, Tom Gundersen, and Michel Parigot. Spinal atomic lambda-calculus. In Jean Goubault-Larrecq and Barbara König, editors, *Foundations of Software Science and Computation Structures (FoSSaCS)*, pages 582–601. Springer International Publishing, 2020.

16 Lutz Straßburger. *Linear Logic and Noncommutativity in the Calculus of Structures*. PhD thesis, Technische Universität Dresden, 2003.

17 Alwen Tiu. A system of interaction and structure II: The need for deep inference. *Logical Methods in Computer Science*, 2(2:4):1–24, April 2006. DOI: 10.2168/LMCS-2(2:4)2006. URL: `https://arxiv.org/pdf/cs.LO/0512036`.

18 A. S. Troelstra and H. Schwichtenberg. *Basic Proof Theory*. Cambridge University Press, 2nd edition, 2000.

A Examples and Explanatory Theorems

▶ **Example 29.** Below is a derivation of the drinker's formula $\exists x.\forall y.\bar{\mathsf{d}}(x)\vee\mathsf{d}(y)$ in SKSq.

$$
\mathsf{m}^{\vee}_{\exists}\,\dfrac{\mathsf{qs}{\downarrow}\,\dfrac{\dfrac{\top}{\forall x.\ \mathsf{qs}{\downarrow}\,\dfrac{\forall y.\ \dfrac{\top}{\mathsf{ai}{\downarrow}\,\dfrac{\top}{\bar{\mathsf{d}}(x)\vee\mathsf{d}(x)}}\,\dagger}{(\forall y.\bar{\mathsf{d}}(x))\vee\dfrac{\exists y.\mathsf{d}(x)}{\exists z.\mathsf{d}(x)}\,\ddagger}}\,\dagger}{(\exists x.\forall y.\bar{\mathsf{d}}(x))\vee\ \mathsf{m}^{\exists}_{\forall}\,\dfrac{\dfrac{\forall x.\exists z.\mathsf{d}(x)}{\forall y.\exists x.\mathsf{d}(y)}\,\ddagger}{\exists x.\forall y.\mathsf{d}(y)}}}{\exists x.\ \mathsf{m}^{\vee}_{\forall}\,\dfrac{(\forall y.\bar{\mathsf{d}}(x))\vee(\forall y.\mathsf{d}(y))}{\forall y.\bar{\mathsf{d}}(x)\vee\mathsf{d}(y)}}
$$

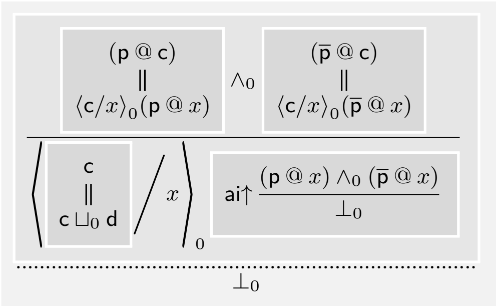
The rule names are written to the left of the rule in vertical compositions as a guide; they are not strictly part of the syntax of derivations. The rules marked † correspond to uses of the equations for vacuous quantification, and the rules marked ‡ correspond to the equations for α-renaming. Note that this proof is contraction-free. Instead, the two rules $\mathsf{m}^{\vee}_{\exists}$ and $\mathsf{m}^{\vee}_{\forall}$ are used to distribute the quantifiers to operands of a disjunction, reminiscent of *miniscoping*. This differs from the sequent calculus, where the only way to prove the drinker's formula is to begin (reading conclusion upwards) with a contraction.

▶ **Proposition 30** (general vacuousness). *The following rules are derivable in* SKSq*, where in each case x is not free in A.*

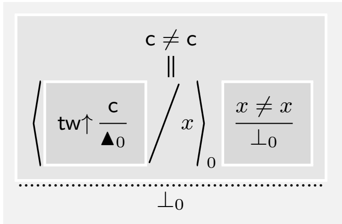
$$
\dfrac{A}{\forall x.\,A} \qquad \dfrac{\exists x.\,A}{A} \qquad \dfrac{\forall x.\,A}{A} \qquad \dfrac{A}{\exists x.\,A}
$$

Proof. The first two rules are derived by induction on the structure of A and using the medials $\mathsf{m}^{\vee}_{\forall}$, $\mathsf{m}^{\wedge}_{\forall}$, $\mathsf{m}^{\exists}_{\vee}$, $\mathsf{m}^{\exists}_{\wedge}$, and $\mathsf{m}^{\exists}_{\forall}$ to distribute the (vacuous) quantifier. The second two are instances of $\mathsf{n}{\uparrow}$ and $\mathsf{n}{\downarrow}$ respectively using any term in $\mathcal{C}_0$ as the witness. ◀

▶ **Example 31.** The problematic examples of Section 4 are fixed as follows, which are both sound.

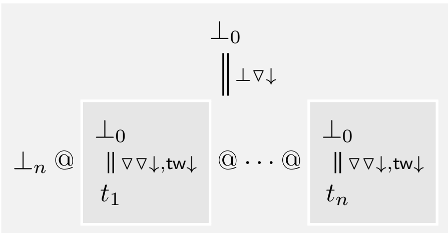


and

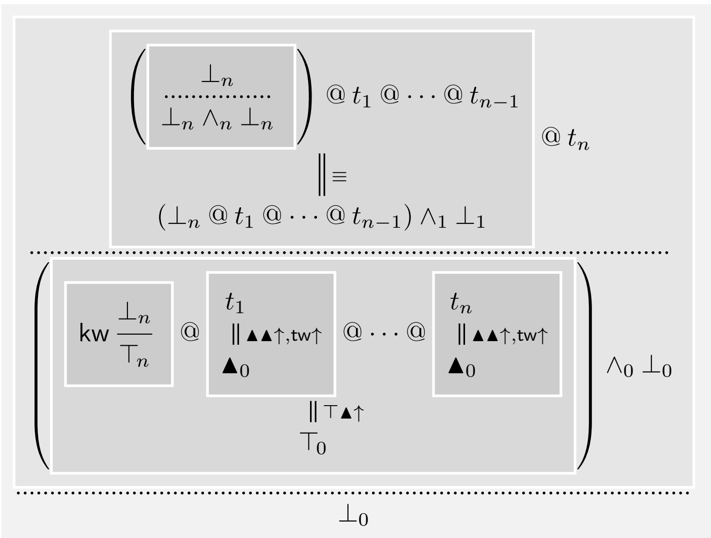


B Proofs in Section 3

Proof of Proposition 18. The first direction, that $\bot_0$ implies $\bot_n \mathbin{@} t_1 \mathbin{@} \cdots \mathbin{@} t_n$, follows from the weakening rules:



The second direction, that $\bot_n \mathbin{@} t_1 \mathbin{@} \cdots \mathbin{@} t_n$ implies $\bot 0$, uses the final equation in Figure 2 first to create $(\bot_n \mathbin{@} t_1 \mathbin{@} \cdots \mathbin{@} t_n) \wedge_0 \bot_0$:



◀

Proof of Proposition 19. The first statement can be proven using the equality $s \equiv s \sqcup_0 \triangledown_0$ and then using term weakening to derive t from $\triangledown_0$. However, to justify the soundness of this equality, we prove the statement without using it as follows:

$$
\mathsf{m}^{\vee}_{@}\,\frac{\begin{array}{c} A \mathbin{@_i} s \\ \hdashline (A \mathbin{@_i} s) \vee_i \left(\bot\triangledown\!\downarrow \dfrac{\bot_{i+1}}{\begin{array}{c}\bot_{i+1} \\ \Big\| {\scriptstyle \bot\triangledown\downarrow,\mathsf{pw}\downarrow,\mathsf{kw},\triangledown\triangledown\downarrow,\mathsf{tw}\downarrow} \\ A\end{array} \mathbin{@_i} \begin{array}{c}\triangledown_0 \\ \Big\| {\scriptstyle \triangledown\triangledown\downarrow,\mathsf{tw}\downarrow} \\ t\end{array}} \right) \end{array}}{\begin{array}{c} A \vee_{i+1} A \\ \Big\| {\scriptstyle \mathsf{m}^{\vee}_{@},\mathsf{pc}\downarrow,\mathsf{m}^{\sqcup}_{@},\mathsf{tc}\downarrow} \\ A \end{array} \mathbin{@_i} (s \sqcup_0 t)}
$$

The other statement has the dual proof using cocontraction and coweakening. ◀

C Proof of Completeness

We recall that the set of ***nonvertical terms***, $\dddot{\mathcal{T}}_i$, are those terms in $\mathcal{T}_i$ without any occurrence of (vertical) composition by inference.

C.1 Contraction

▶ **Lemma 32.** *For all $t \in \dddot{\mathcal{T}}_i$, there is a derivation* $\begin{array}{c} t \sqcup_i t \\ \Phi \Big\| \mathsf{KSqt} \\ t \end{array}$.

Proof. We proceed by induction on the structure of t.

- If $t \in \mathcal{C}_i \cup \mathcal{V}_i \cup \{\triangledown_i, \blacktriangle_i\}$ then $\Phi = \mathsf{tc}\!\downarrow \dfrac{t \sqcup_i t}{t}$.
- If t is $s \mathbin{@_T} q$, for $s \in \dddot{\mathcal{T}}_{i+1}$ and $q \in \dddot{\mathcal{T}}_0$, then by induction we can build:

$$
\Phi = \mathsf{m}^{\vee}_{@}\,\frac{(s \mathbin{@_T} q) \sqcup_i (s \mathbin{@_T} q)}{\begin{array}{c} s \sqcup_{i+1} s \\ \mathsf{IH} \Big\| \mathsf{KSqt} \\ s \end{array} \mathbin{@} \begin{array}{c} q \sqcup_0 q \\ \mathsf{IH} \Big\| \mathsf{KSqt} \\ q \end{array}}
$$

- If $t = s \sqcup_i q$, for $s, q \in \dddot{\mathcal{T}}_i$, then by induction and associativity of $\sqcup_i$ we can build:

$$
\Phi = \begin{array}{c} (s \sqcup_i q) \sqcup i (s \sqcup_i q) \\ \hdashline \begin{array}{c} s \sqcup_i s \\ \mathsf{IH} \Big\| \mathsf{KSqt} \\ s \end{array} \sqcup_i \begin{array}{c} q \sqcup_i q \\ \mathsf{IH} \Big\| \mathsf{KSqt} \\ q \end{array} \end{array}
$$

- If $t = s \sqcap_i q$, for $s, q \in \dddot{\mathcal{T}}_i$, then by induction we can build:

$$\Phi = \mathsf{m}^{\sqcup}_{\sqcap}\,\frac{(s \sqcap_i q) \sqcup_i (s \sqcap_i q)}{\boxed{\begin{array}{c} s \sqcup_i s \\ \mathsf{IH} \,\Vert\, \mathsf{KSqt} \\ s \end{array}} \sqcup i \boxed{\begin{array}{c} q \sqcup_i q \\ \mathsf{IH} \,\Vert\, \mathsf{KSqt} \\ q \end{array}}}$$

- If $t = \langle s/x \rangle_j\, q$, for $s \in \dddot{\mathcal{T}}_j$ and $q \in \dddot{\mathcal{T}}_i$, then by induction we can build:

$$\Phi = \mathsf{m}^{\sqcup}_{\sigma}\,\frac{\langle s/x \rangle_j\, q \sqcup_i \langle s/x \rangle_j\, q}{\left\langle \boxed{\begin{array}{c} s \sqcup_j s \\ \mathsf{IH} \,\Vert\, \mathsf{KSqt} \\ s \end{array}} \Big/ x \right\rangle_j \boxed{\begin{array}{c} q \sqcup_i q \\ \mathsf{IH} \,\Vert\, \mathsf{KSqt} \\ q \end{array}}}$$

◀

▶ **Lemma 33.** *For all $A \in \dddot{\mathcal{A}}_i$, there is a derivation* $\begin{array}{c} A \vee_i A \\ \Phi \,\Vert\, \mathsf{SKSqt} \\ A \end{array}$.

Proof. We proceed by induction on the structure of A.

- If $A \in \mathcal{L}_i$ then $\Phi = \mathsf{pc}{\downarrow}\,\dfrac{A \vee_i A}{A}$.
- If A is $\bot_i$ or $\top_i$ then Φ is an equality ($\equiv$).
- If $A = B \mathbin{@_i} t$, for $B \in \dddot{\mathcal{A}}_{i+1}$ and $t \in \dddot{\mathcal{T}}_0$, then by Lemma 32 and induction we can build:

$$\Phi = \mathsf{m}^{\vee}_{@}\,\frac{(B \mathbin{@_i} t) \vee_i (B \mathbin{@_i} t)}{\boxed{\begin{array}{c} B \vee_{i+1} B \\ \mathsf{IH} \,\Vert\, \mathsf{KSqt} \\ B \end{array}} \mathbin{@_i} \boxed{\begin{array}{c} t \sqcup_0 t \\ \mathsf{Lem.32} \,\Vert\, \mathsf{KSqt} \\ t \end{array}}}$$

- If $A = B \vee_i C$, for $B, C \in \dddot{\mathcal{A}}_i$, then by associativity and induction we can build:

$$\Phi = \frac{(B \vee_i C) \vee_i (B \vee_i C)}{\boxed{\begin{array}{c} B \vee_i B \\ \mathsf{IH} \,\Vert\, \mathsf{KSqt} \\ B \end{array}} \vee_i \boxed{\begin{array}{c} C \vee_i C \\ \mathsf{IH} \,\Vert\, \mathsf{KSqt} \\ C \end{array}}}$$

- If $A = B \wedge_i C$, for $B, C \in \dddot{\mathcal{A}}_i$ then by induction we can build:

$$\Phi = \mathsf{m}\,\frac{(B \wedge_i C) \vee_i (B \wedge_i C)}{\boxed{\begin{array}{c} B \vee_i B \\ \mathsf{IH} \,\Vert\, \mathsf{KSqt} \\ B \end{array}} \wedge_i \boxed{\begin{array}{c} C \vee_i C \\ \mathsf{IH} \,\Vert\, \mathsf{KSqt} \\ C \end{array}}}$$

- If $A = \langle t/x \rangle_j\, B$, for $t \in \dddot{\mathcal{T}}_j$ and $B \in \dddot{\mathcal{A}}_i$ and $i > 0$, then by Lemma 32 and induction we can build:

$$\Phi = \mathsf{m}^{\vee}_{\sigma}\,\frac{\langle t/x \rangle_j\, B \vee_i \langle t/x \rangle_j\, B}{\left\langle \boxed{\begin{array}{c} t \sqcup_j t \\ \mathsf{Lem.32} \,\Vert\, \mathsf{KSqt} \\ t \end{array}} \Big/ x \right\rangle_j \boxed{\begin{array}{c} B \vee_i B \\ \mathsf{IH} \,\Vert\, \mathsf{KSqt} \\ B \end{array}}}$$

  The case for $i = 0$ is similar, with $A = \langle t/x \rangle_j\, B$ and $\mathsf{m}^{\vee_0}_{\sigma}$ used in place of $\mathsf{m}^{\vee}_{\sigma}$.
- If $A = (B \leftrightharpoons_i C)$, for $B, C \in \dddot{\mathcal{A}}_i$, then we build:

$$\Phi = \mathsf{c}^{\leftrightharpoons}{\downarrow}\frac{(B \leftrightharpoons_i C) \vee_0 (B \leftrightharpoons_i C)}{\begin{array}{c} B \vee_i B \\ \mathsf{IH}\ \Vert\ \mathsf{KSqt} \\ B \end{array} \leftrightharpoons_i \begin{array}{c} C \vee_i C \\ \mathsf{IH}\ \Vert\ \mathsf{KSqt} \\ C \end{array}}$$

The case for $A = (B \not\leftrightharpoons_i C)$ is similar, using $\mathsf{c}^{\not\leftrightharpoons}{\downarrow}$ in place of $\mathsf{c}^{\leftrightharpoons}{\downarrow}$.

- If $A = (s \circeq_i t)$, for $s, t \in \ddot{\mathcal{T}}_i$, then we build:

$$\Phi = \mathsf{c}^{\circeq}{\downarrow}\frac{(s \circeq_i t) \vee_0 (s \circeq_i t)}{\begin{array}{c} s \sqcup_i s \\ \mathsf{Lem.32}\ \Vert\ \mathsf{KSqt} \\ s \end{array} \circeq_i \begin{array}{c} t \sqcup_i t \\ \mathsf{Lem.32}\ \Vert\ \mathsf{KSqt} \\ t \end{array}}$$

The case for $A = (s \not\circeq_i t)$ is similar, using $\mathsf{c}^{\not\circeq}{\downarrow}$ in place of $\mathsf{c}^{\circeq}{\downarrow}$. ◀

▶ **Corollary 34.** $\mathsf{ac}{\downarrow}$ *is derivable in* KSqt *and* $\mathsf{ac}{\uparrow}$ *is derivable in* SKSqt.

C.2 Weakening

▶ **Lemma 35.** *For all* $t \in \ddot{\mathcal{T}}_i$*, there is a derivation* $\begin{array}{c} \triangledown_i \\ \Phi\ \Vert\ \mathsf{KSqt} \\ t \end{array}$.

Proof. We proceed by induction on the structure of t.

- If $t \in \mathcal{C}_i \cup \mathcal{V}_i \cup \{\blacktriangle_i\}$ then $\Phi = \mathsf{tw}{\downarrow}\frac{\triangledown_i}{t}$. If $t = \triangledown_i$ then $\Phi = t$.
- If $t = (s \mathbin{@_T} q)$, for $s \in \ddot{\mathcal{T}}_{i+1}$ and $q \in \ddot{\mathcal{T}}_0$, then by induction we can build:

$$\Phi = \triangledown\triangledown{\downarrow}\frac{\triangledown_i}{\begin{array}{c} \triangledown_{i+1} \\ \mathsf{IH}\ \Vert\ \mathsf{KSqt} \\ s \end{array} @_T \begin{array}{c} \triangledown_0 \\ \mathsf{IH}\ \Vert\ \mathsf{KSqt} \\ q \end{array}}$$

- If $t = s *_i q$, for $s, q \in \ddot{\mathcal{T}}_i$ and $* \in \{\sqcup, \sqcap\}$, then by induction we can build:

$$\begin{array}{c} \triangledown_i \\ \hline \begin{array}{c} \triangledown_i \\ \mathsf{IH}\ \Vert\ \mathsf{KSqt} \\ s \end{array} *_i \begin{array}{c} \triangledown_i \\ \mathsf{IH}\ \Vert\ \mathsf{KSqt} \\ q \end{array} \end{array}$$

- If $t = \langle s/x \rangle_j\, q$, for $s \in \ddot{\mathcal{T}}_j$ and $q \in \ddot{\mathcal{T}}_i$, then by induction we can build:

$$\mathsf{v}\sigma{\downarrow}\frac{\triangledown_i}{\left\langle \begin{array}{c} \triangledown_j \\ \mathsf{IH}\ \Vert\ \mathsf{KSqt} \\ s \end{array} \middle/ x \right\rangle_j \begin{array}{c} \triangledown_i \\ \mathsf{IH}\ \Vert\ \mathsf{KSqt} \\ q \end{array}}$$

◀

▶ **Lemma 36.** *For all* $A \in \ddot{\mathcal{A}}_i$*, there is a derivation* $\begin{array}{c} \bot_i \\ \Phi\ \Vert\ \mathsf{KSqt} \\ A \end{array}$.

Proof. We proceed by induction on the structure of A.

- If $A \in \mathcal{L}_i$ then $\Phi = \mathsf{pw}{\downarrow}\dfrac{\bot_i}{A}$.
- If $A = \bot_i$ then Φ is trivial (i.e., $\Phi = \bot_i$). If $A = \top_i$ then we can build

$$\Phi = \mathsf{m}\,\dfrac{\dfrac{\bot_i}{\left[\dfrac{\bot_i}{\bot_i \wedge_i \top_i}\right] \vee_i \left[\dfrac{\bot_i}{\top_i \wedge_i \bot_i}\right]}}{\dfrac{\left[\dfrac{\bot_i \vee_i \top_i}{\top_i}\right] \vee_i \left[\dfrac{\top_i \vee_i \bot_i}{\top_i}\right]}{\top_i}}$$

- If $A = B \mathbin{@_i} t$, for $B \in \ddot{\mathcal{A}}_{i+1}$ and $t \in \ddot{\mathcal{T}}_0$, then by Lemma 35 and induction we can build:

$$\Phi = \bot\triangledown{\downarrow}\,\dfrac{\bot_i}{\left[\begin{array}{c}\bot_{i+1}\\ \mathsf{IH}\,\Vert\,\mathsf{KSqt}\\ B\end{array}\right] @_i \left[\begin{array}{c}\triangledown_0\\ \mathsf{Lem.35}\,\Vert\,\mathsf{KSqt}\\ t\end{array}\right]}$$

- If $A = B *_i C$, for $B, C \in \ddot{\mathcal{A}}_i$ and $* \in \{\vee, \wedge\}$ then by induction we can build

$$\Phi = \dfrac{\bot_i}{\left[\begin{array}{c}\bot_i\\ \mathsf{IH}\,\Vert\,\mathsf{KSqt}\\ B\end{array}\right] *_i \left[\begin{array}{c}\bot_i\\ \mathsf{IH}\,\Vert\,\mathsf{KSqt}\\ C\end{array}\right]}$$

- If $A = \langle t/x \rangle_j B$ then by Lemma 35 and induction we can build:

$$\Phi = \dfrac{\bot_i}{\left\langle \left[\begin{array}{c}\triangledown_j\\ \mathsf{Lem.35}\,\Vert\,\mathsf{KSqt}\\ t\end{array}\right] \Big/ x \right\rangle_j \left[\begin{array}{c}\bot_i\\ \mathsf{IH}\,\Vert\,\mathsf{KSqt}\\ B\end{array}\right]}$$

- If $A = (B \leftrightharpoons_i C)$, for $B, C \in \ddot{\mathcal{A}}_i$, then by Lemma 36 we can build

$$\Phi = \mathsf{w}^{\leftrightharpoons}{\downarrow}\,\dfrac{\bot_0}{\left[\begin{array}{c}\bot_i\\ \mathsf{Lem.36}\,\Vert\,\mathsf{KSqt}\\ B\end{array}\right] \leftrightharpoons_i \left[\begin{array}{c}\bot_i\\ \mathsf{Lem.36}\,\Vert\,\mathsf{KSqt}\\ C\end{array}\right]}$$

  The case for $\not\leftrightharpoons_i$ is similar, using $\mathsf{w}^{\not\leftrightharpoons}{\downarrow}$ in place of $\mathsf{w}^{\leftrightharpoons}{\downarrow}$.
- If $A = (s \doteq_i t)$, for $s, t, \in \ddot{\mathcal{T}}_i$, then by Lemma 35 we can build

$$\Phi = \mathsf{w}^{\doteq}{\downarrow}\,\dfrac{\bot_0}{\left[\begin{array}{c}\triangledown_i\\ \mathsf{Lem.35}\,\Vert\,\mathsf{KSqt}\\ s\end{array}\right] \doteq_i \left[\begin{array}{c}\triangledown_i\\ \mathsf{Lem.35}\,\Vert\,\mathsf{KSqt}\\ t\end{array}\right]}$$

  The case for $\not\doteq_i$ is similar, using $\mathsf{w}^{\not\doteq}{\downarrow}$ in place of $\mathsf{w}^{\doteq}{\downarrow}$. ◀

▶ **Corollary 37.** $\mathsf{aw}{\downarrow}$ *is derivable in* KSqt *and* $\mathsf{aw}{\uparrow}$ *is derivable in* SKSqt. ◀

C.3 Instantiation

We recall that $\mathcal{E}_i$ denotes the subset of $\mathcal{T}_i$ consisting only of elementary terms composed by no more than a single composition by application, so that $t \in \mathcal{E}_i$ is either of the form d or $\mathsf{f} \mathbin{@_T} k_c$, for $\mathsf{d}, \mathsf{f}, \mathsf{c} \in \mathcal{C} \cup \mathcal{V}$.

▶ **Definition 38.** *Let $\dddot{\mathcal{T}}_i^{@}$ denote the subset of $\dddot{\mathcal{T}}_i$ consisting only of those elements composed by constants, variables and application, so that $t \in \dddot{\mathcal{T}}_i^{@}$ doesn't contain any occurrences of $\nabla_i, \blacktriangle_i, \sqcup_i, \sqcap_i$, or explicit substitution. For $t \in \dddot{\mathcal{T}}_i^{@}$, we define its* **substitution decomposition into** x*, denoted by $\sigma^t_{x,i}$, to be the compound explicit substitution constructed as follows:*

- *If $t \in \mathcal{C}_i \cup \mathcal{V}_i$ then $\sigma^t_{x,i} = \langle t/x \rangle_i$*
- *If $t = f \mathbin{@_T} s$, for $f \in \dddot{\mathcal{T}}_{i+1}^{@}$ and $s \in \dddot{\mathcal{T}}_0^{@}$, then $\sigma^t_{x,i} = \sigma^f_{y,i+1}\ \sigma^s_{z,0}\ \langle y \mathbin{@_T} z/x \rangle_i$*

▶ **Lemma 39.** *Let $t \in \dddot{\mathcal{T}}_i^{@}$.*

- *There exists a derivation* $\begin{array}{c} \sigma^t_{x,i} \\ \Phi \,\|\, \mathsf{KSqt} \\ \exists_i x \end{array}$
- *For any $A \in \mathcal{F}$ there exists a derivation* $\begin{array}{c} \tau_1 \cdots \tau_n\ A \\ \Omega \,\|\, \mathsf{KSqt} \\ \sigma^t_{x,i}\ A \end{array}$ *where for all $j \in 1..n$, the substitution term of τ_j is elementary.*

Proof. We proceed by induction on the structure of t.

- If $t \in \mathcal{C}_i \cup \mathcal{V}_i$ then $\Phi = \mathsf{qn}{\downarrow}\dfrac{\langle c/x \rangle_i}{\exists_i x}$ and for any $A \in \mathcal{F}$, $\Omega = \langle c_i/x_i \rangle\, A$.
- If $t = s \mathbin{@_T} q$, for $s \in \dddot{\mathcal{T}}_{i+1}^{@}$ and $q \in \dddot{\mathcal{T}}_0^{@}$ then by induction we can build:

$$\Phi = \mathsf{x}{\downarrow}\dfrac{\begin{array}{c} \sigma^s_{y,i+1} \\ \mathsf{IH} \,\|\, \mathsf{KSqt} \\ \exists_{i+1} y \end{array} \quad \begin{array}{c} \sigma^q_{z,0} \\ \mathsf{IH} \,\|\, \mathsf{KSqt} \\ \exists_0 z \end{array} \quad \langle y \mathbin{@_T} z/x \rangle_i}{\exists_i x}$$

where y and z are fresh variables. For any formula A, by induction we can build:

$$\Omega = \begin{array}{c} \tau_1 \ \cdots\ \tau_k\ \tau_{k+1}\ \cdots\ \tau_n\ \langle y \mathbin{@_T} z/x \rangle_i\ A \\ \mathsf{IH} \,\|\, \mathsf{KSqt} \\ \sigma^s_{y,i+1} \ \left[\begin{array}{c} \tau_{k+1}\ \cdots\ \tau_n\ \langle y \mathbin{@_T} z/x \rangle_i\ A \\ \mathsf{IH} \,\|\, \mathsf{KSqt} \\ \sigma^q_{z,0}\ \langle y \mathbin{@_T} z/x \rangle_i\ A \end{array}\right] \end{array}$$

◀

▶ **Lemma 40.** *For any $s \in \mathcal{E}_i$ and any $t \in \dddot{\mathcal{T}}_j$, there is a derivation* $\begin{array}{c} [s/x]_i\, t \\ \Phi \,\|\, \mathsf{KSqt} \\ \langle s/x \rangle_i\, t \end{array}$.

Proof. We proceed by induction on the structure of t.

- If $t = x$ then $\Phi = \sigma{\downarrow}\dfrac{s}{\langle s/x \rangle_i\, t}$.

- If $t \in \mathcal{C}_j \cup \mathcal{V}_j \cup \{\triangledown_j, \blacktriangle_j\}$ and $t \neq x$ then $\Phi = \begin{array}{c} t \\ \cdots\cdots \\ \langle s/x \rangle_i\, t \end{array}$.
- If $t = r \mathbin{@_T} q$, for $r \in \dddot{\mathcal{T}}_{j+1}$ and $q \in \dddot{\mathcal{T}}_0$, then by induction we can build:

$$\Phi = \sigma@_T\!\downarrow \dfrac{\begin{array}{c} [s/x]\, r \\ \mathsf{IH} \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, r \end{array} \;@_T\; \begin{array}{c} [s/x]_i\, q \\ \mathsf{IH} \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, q \end{array}}{\langle s/x \rangle_i (s \mathbin{@_T} q)}$$

- If $t = r \sqcup_j q$, for $r, q \in \dddot{\mathcal{T}}_j$, then by induction we can build:

$$\Phi = \mathsf{m}^{\sqcup}_{\sigma} \dfrac{\begin{array}{c} [s/x]_i\, r \\ \mathsf{IH} \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, r \end{array} \;\sqcup_j\; \begin{array}{c} [s/x]_i\, q \\ \mathsf{IH} \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, q \end{array}}{\left\langle \begin{array}{c} s \sqcup_i s \\ \mathsf{Lem.32} \Vert \mathsf{KSqt} \\ s \end{array} \middle/ x \right\rangle_i (r \sqcup_i q)}$$

- If $t = r \sqcap_j q$, for $r, q \in \dddot{\mathcal{T}}_j$, then by induction we can build:

$$\Phi = \sigma\sqcap\!\downarrow \dfrac{\begin{array}{c} [s/x]_i\, r \\ \mathsf{IH} \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, r \end{array} \;\sqcap_j\; \begin{array}{c} [s/x]_i\, q \\ \mathsf{IH} \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, q \end{array}}{\langle s/x \rangle_i (r \sqcap_j q)}$$

- If $t = \langle r/y \rangle_k\, q$, for $r \in \dddot{\mathcal{T}}_k$, $q \in \dddot{\mathcal{T}}_j$, and $x_i \neq y_k$, then by induction we can build:

$$\Phi = \sigma\sigma\!\downarrow \dfrac{\left\langle \begin{array}{c} [s/x]_i\, r \\ \mathsf{IH} \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, r \end{array} \middle/ y \right\rangle_k \begin{array}{c} [s/x]_i\, q \\ \mathsf{IH} \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, q \end{array}}{\langle s/x \rangle_i\, \langle r/y \rangle_k\, q}$$

◀

▶ **Lemma 41.** *For any $s \in \mathcal{E}_i$ and any $A \in \dddot{\mathcal{A}}_j$, there is a derivation* $\begin{array}{c} [s/x]_i\, A \\ \Phi \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, A \end{array}$.

Proof. We proceed by induction on the structure of A.

- If $A \in \mathcal{L}_i \cup \{\bot_i, \top_i\}$ then $\Phi = \begin{array}{c} A \\ \cdots\cdots \\ \langle s/x \rangle_i\, A \end{array}$.
- If $A = B \mathbin{@_i} t$, for $B \in \mathcal{F}^{\mathcal{P}}_{i+1}$ and $t \in \dddot{\mathcal{T}}_0$, then by induction and Lemma 40, we can build:

$$\Phi = \sigma@\!\downarrow \dfrac{\begin{array}{c} [s/x]_i\, B \\ \mathsf{IH} \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, B \end{array} \;@_i\; \begin{array}{c} [s/x]_i\, t \\ \mathsf{IH} \Vert \mathsf{KSqt} \\ \langle s/x \rangle_i\, t \end{array}}{\langle s/x \rangle_i (B \mathbin{@_i} t)}$$

- If $A = B \vee_i C$, for $B, C \in \dddot{\mathcal{A}}_i$, then by induction we can build:

$$\Phi = \mathsf{m}^{\vee}_{\sigma}\,\frac{\begin{array}{c}[s/x]_i\, B \\ \mathsf{IH}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, B\end{array} \;\vee_i\; \begin{array}{c}[s/x]_i\, C \\ \mathsf{IH}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, C\end{array}}{\left\langle \mathsf{tc}{\downarrow}\,\dfrac{d_i \sqcup_i d_i}{s} \middle/ x \right\rangle_i (B \vee_i C)}$$

- If $A = B \wedge_i C$, for $B, C \in \dddot{\mathcal{A}}i$, then by induction we can build:

$$\Phi = \sigma{\wedge}{\downarrow}\,\frac{\begin{array}{c}[s/x]_i\, B \\ \mathsf{IH}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, B\end{array} \;\wedge_i\; \begin{array}{c}[s/x]_i\, C \\ \mathsf{IH}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, C\end{array}}{\langle s/x\rangle_i (B \wedge_i C)}$$

- If $A = \langle q/y\rangle_k\, B$, for $q \in \dddot{\mathcal{T}}_k$, $B \in \dddot{\mathcal{A}}_i$, and $x_i \neq y_k$, then by induction we can build:

$$\Phi = \sigma\sigma{\downarrow}\,\frac{\left\langle \begin{array}{c}[s/x]_i\, q \\ \mathsf{IH}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, q\end{array} \middle/ y \right\rangle_k \begin{array}{c}[s/x]_i\, B \\ \mathsf{IH}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, B\end{array}}{\langle s/x\rangle_i\, \langle q/y\rangle_k\, B}$$

- If $A = (B \fallingdotseq_i C)$, for $B, C \in \dddot{\mathcal{A}}_i$, then by induction we can build:

$$\Phi = 2{\times}\sigma@{\downarrow}\,\frac{\begin{array}{c}[s/x]_i\, B \\ \mathsf{IH}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, B\end{array} \;\fallingdotseq_i\; \begin{array}{c}[s/x]_i\, C \\ \mathsf{IH}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, C\end{array}}{\langle s/x\rangle_i (B \fallingdotseq_i C)}$$

  where $2{\times}\sigma@{\downarrow}$ denotes two instances of the rule $\sigma@{\downarrow}$ because we read $B \fallingdotseq_i C$ as $(\fallingdotseq_i @_1 B)\, @_0\, C$. The case for $A = (B \not\fallingdotseq_i C)$ is similar.
- If $A = (q \mathrel{\mathring{=}}_i t)$, for $q, t \in \dddot{\mathcal{T}}_i$, then by Lemma 40 we can build:

$$\Phi = 2{\times}\sigma@{\downarrow}\,\frac{\begin{array}{c}[s/x]_i\, q \\ \mathsf{Lem.40}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, q\end{array} \;\mathring{=}_i\; \begin{array}{c}[s/x]_i\, t \\ \mathsf{Lem.40}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, t\end{array}}{\langle s/x\rangle_i (q \mathrel{\mathring{=}}_i t)}$$

  The case for $A = (s \mathrel{\mathring{\neq}}_i t)$ is similar. ◀

▶ **Lemma 42.** *For any $t \in \mathcal{E}_i$ and any $A \in \mathcal{F}$, there is a derivation* $\begin{array}{c}[s/x]_i\, A \\ \Phi\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, A\end{array}$.

Proof. We proceed by induction on the structure of A.

- If $A \in \dddot{\mathcal{A}}_0$ then the result follows from Lemma 41.
- If $A = B \vee C$ or $A = B \wedge C$ then we proceed by induction as in the cases for $\vee_i$ or $\wedge_i$ in Lemma 41.
- If $A = \mathsf{Q}_j y.\, B$, for $\mathsf{Q} \in \{\exists, \forall\}$ and $B \in \mathcal{F}$, then by induction we can build:

$$\Phi = \sigma\mathsf{Q}{\downarrow}\,\frac{\mathsf{Q}_j y.\ \begin{array}{c}[s/x]_i\, B \\ \mathsf{IH}\,\Vert\,\mathsf{KSqt} \\ \langle s/x\rangle_i\, B\end{array}}{\langle s/x\rangle_i\, \mathsf{Q}_j y.\, B}$$

- If $A = \langle t/y \rangle_j B$, for $t \in \dddot{\mathcal{T}}_j$ and $B \in \mathcal{F}$, then by induction and Lemma 40 we can build:

$$\Phi = \quad \sigma\sigma\!\downarrow \frac{\left\langle \begin{array}{c} [s/x]_i\, t \\ \scriptstyle \text{Lem.40}\, \Vert\, \mathsf{KSqt} \\ \langle s/x \rangle_i\, t \end{array} \Big/ y \right\rangle_j \begin{array}{c} [s/x]_i\, B \\ \scriptstyle \mathsf{IH}\, \Vert\, \mathsf{KSqt} \\ \langle s/x \rangle_i\, B \end{array}}{\langle s/x \rangle_i \langle t/y \rangle_j B}$$

◀

▶ **Corollary 43.** $\mathsf{n}\!\downarrow$ *is derivable in* KSqt *and* $\mathsf{n}\!\uparrow$ *is derivable in* SKSqt. ◀

C.4 Vacuous Quantification

▶ **Definition 44.** *In the following,* $\mathsf{S}(W, x)$ *stands for* KSqt *if* x *does not appear free in* W *and* SKSqt *otherwise.*

▶ **Lemma 45.** *For any* $t \in \dddot{\mathcal{T}}_j$ *there is a derivation* $\begin{array}{c} \langle \blacktriangle_i/x \rangle_i\, t \\ \scriptstyle \Phi\, \Vert\, \mathsf{S}(t,x) \\ [\blacktriangle_i/x]_i\, t \end{array}$.

Proof. We proceed by induction on the structure of t.

- If $t = x_i$ then $\Phi = \quad \blacktriangle\mathsf{v}\!\downarrow \dfrac{\langle \blacktriangle_i/x \rangle_i \quad \mathsf{tw}\!\uparrow \dfrac{x_i}{\blacktriangle_i}}{\blacktriangle_i}$

- If $t \in \mathcal{C}_j \cup \mathcal{V}_j \cup \{\triangledown_j, \blacktriangle_j\}$, then $\Phi = \quad \blacktriangle\mathsf{v}\!\downarrow \dfrac{\langle \blacktriangle_i/x \rangle_i\, t}{t}$.

- If $t = s *_j q$, for $* \in \{@, \sqcup, \sqcap\}$ and $s \in \dddot{\mathcal{T}}_k$ and $q \in \dddot{\mathcal{T}}_k$, where k, l can be determined from j and $*$, then

$$\Phi = \quad \blacktriangle{*}\!\downarrow \frac{\langle \blacktriangle_i/x \rangle_i (s *_j q)}{\begin{array}{c} \langle \blacktriangle_i/x \rangle_i\, s \\ \scriptstyle \mathsf{IH}\, \Vert\, \mathsf{S}(s,x) \\ [\blacktriangle_i/x]_i\, s \end{array} \; *_j \; \begin{array}{c} \langle \blacktriangle_i/x \rangle_i\, q \\ \scriptstyle \mathsf{IH}\, \Vert\, \mathsf{S}(q,x) \\ [\blacktriangle_i/x]_i\, q \end{array}}$$

- If $t = \langle s/y \rangle_j q$, for $s \in \dddot{\mathcal{T}}_j$ and $q \in \dddot{\mathcal{T}}_i$, then

$$\Phi = \quad \blacktriangle\sigma\!\downarrow \frac{\langle \blacktriangle_i/x \rangle_i \langle s/y \rangle_j\, q}{\left\langle \begin{array}{c} \langle \blacktriangle_i/x \rangle_i\, s \\ \scriptstyle \mathsf{IH}\, \Vert\, \mathsf{S}(s,x) \\ [\blacktriangle_i/x]_i\, s \end{array} \Big/ y \right\rangle_j \begin{array}{c} \langle \blacktriangle_i/x \rangle_i\, q \\ \scriptstyle \mathsf{IH}\, \Vert\, \mathsf{S}(q,x) \\ [\blacktriangle_i/x]_i\, q \end{array}}$$

◀

▶ **Lemma 46.** *For any* $A \in \ddot{\mathcal{A}}_j$ *there is a derivation* $\begin{array}{c} \langle \blacktriangle_i/x \rangle_i\, A \\ \scriptstyle \Phi\, \Vert\, \mathsf{S}(A,x) \\ [\blacktriangle_i/x]_i\, A \end{array}$.

Proof. We proceed by induction on the structure of A.

- If $A \in \mathcal{L}_i \cup \{\bot_i, \top_i\}$ then $\Phi = \blacktriangle\mathsf{v}{\downarrow}\dfrac{\langle \blacktriangle_i/x\rangle_i\, A}{A}$.
- If $A = B \mathbin{@_j} t$, for $B \in \ddot{\mathcal{A}}_{j+1}$ and $t \in \ddot{\mathcal{T}}_0$, by induction and Lemma 45 we can build:

$$\Phi = \blacktriangle@{\downarrow}\dfrac{\langle \blacktriangle_i/x\rangle_i(B \mathbin{@_j} t)}{\begin{array}{c}\langle \blacktriangle_i/x\rangle_i\, B\\ \mathsf{IH}\,\Vert\,\mathsf{S}(B,x)\\ [\blacktriangle_i/x]_i\, B\end{array} \mathbin{@_j} \begin{array}{c}\langle \blacktriangle_i/x\rangle_i\, t\\ \mathsf{Lem.45}\,\Vert\,\mathsf{S}(t,x)\\ [\blacktriangle_i/x]_i\, t\end{array}}$$

- If $A = B *_j C$, for $* \in \{\vee, \wedge\}$ and $B, C \in \ddot{\mathcal{A}}_j$, by induction we can build:

$$\Phi = \blacktriangle{*}{\downarrow}\dfrac{\langle \blacktriangle_i/x\rangle_i(B *_j C)}{\begin{array}{c}\langle \blacktriangle_i/x\rangle_i\, B\\ \mathsf{IH}\,\Vert\,\mathsf{S}(B,x)\\ [\blacktriangle_i/x]_i\, B\end{array} *_j \begin{array}{c}\langle \blacktriangle_i/x\rangle_i\, C\\ \mathsf{IH}\,\Vert\,\mathsf{S}(C,x)\\ [\blacktriangle_i/x]_i\, C\end{array}}$$

- If $A = \langle s/y\rangle_k\, B$, for $s \in \ddot{\mathcal{T}}_k$ and $B \in \ddot{\mathcal{A}}j$, by induction and Lemma 45 we can build:

$$\Phi = \blacktriangle\sigma{\downarrow}\dfrac{\langle \blacktriangle_i/x\rangle_i\, \langle s/y\rangle_k\, B}{\left\langle \begin{array}{c}\langle \blacktriangle_i/x\rangle_i\, s\\ \mathsf{Lem.45}\,\Vert\,\mathsf{S}(s,x)\\ [\blacktriangle_i/x]_i\, s\end{array} \middle/ y \right\rangle_k \begin{array}{c}\langle \blacktriangle_i/x\rangle_i\, B\\ \mathsf{IH}\,\Vert\,\mathsf{S}(B,x)\\ [\blacktriangle_i/x]_i\, B\end{array}}$$

If $A = (B \leftrightharpoons_i C)$, for $B, C \in \ddot{\mathcal{A}}_i$, then we can build

$$\Phi = 2{\times}\blacktriangle@{\downarrow}\dfrac{\langle \blacktriangle_i/x\rangle_i(B \leftrightharpoons_i C)}{\begin{array}{c}\langle \blacktriangle_i/x\rangle_i\, B\\ \mathsf{Lem.46}\,\Vert\,\mathsf{KSqt}\\ [\blacktriangle_i/x]_i\, B\end{array} \leftrightharpoons_i \begin{array}{c}\langle \blacktriangle_i/x\rangle_i\, C\\ \mathsf{Lem.46}\,\Vert\,\mathsf{KSqt}\\ [\blacktriangle_i/x]_i\, C\end{array}}$$

The cases for $B \not\leftrightharpoons_i C$, $s \circeq_i t$, and $s \not\circeq_i t$ are similar, with the latter two following from Lemma 45. ◀

▶ **Lemma 47.** *For all $A \in \mathcal{F}$ there is a derivation* $\begin{array}{c}\exists_i x.\, A\\ \Phi\,\Vert\,\mathsf{S}(A,x)\\ [\blacktriangle_i/x]_i\, A\end{array}$.

Proof. We proceed by induction on the structure of A.

- If $A \in \ddot{\mathcal{A}}_0$ then we can build:

$$\Phi = \begin{array}{c}\left(\mathsf{qv}{\downarrow}\dfrac{\exists_i x}{\langle \blacktriangle_i/x_i\rangle}\right) A\\ \mathsf{Lem.46}\,\Vert\,\mathsf{S}(A,x)\\ [\blacktriangle_i/x]_i\, A\end{array}$$

- If $A = B \vee C$, for $B, C \in \mathcal{F}$, then we can build:

$$\Phi = \mathsf{m}^{\exists}_{\vee}\dfrac{\exists_i x.\,(B \vee C)}{\begin{array}{c}\exists_i x.\, B\\ \mathsf{IH}\,\Vert\,\mathsf{S}(B,x)\\ [\blacktriangle_i/x]_i\, B\end{array} \vee \begin{array}{c}\exists_i x.\, C\\ \mathsf{IH}\,\Vert\,\mathsf{S}(C,x)\\ [\blacktriangle_i/x]_i\, C\end{array}}$$

- If $A = B \wedge C$, for $B, C \in \mathcal{F}$, then we can build:

$$\Phi = \mathsf{m}^{\exists}_{\wedge}\,\frac{\exists_i x.\,(B\wedge C)}{\begin{array}{c}\exists_i x.\,B\\ \mathsf{IH}\,\|\,\mathsf{S}(B,x)\\ {[\blacktriangle_i/x]_i}\,B\end{array}\ \wedge\ \begin{array}{c}\exists_i x.\,C\\ \mathsf{IH}\,\|\,\mathsf{S}(C,x)\\ {[\blacktriangle_i/x]_i}\,C\end{array}}$$

- If $A = \exists_j y.\,B$, for $B \in \mathcal{F}$, then we can build:

$$\Phi = \begin{array}{c}\exists_i x.\,\exists_j y.\,B\\ \cdots\cdots\cdots\cdots\\ \exists_j y\ \begin{array}{c}\exists_i x.\,B\\ \mathsf{IH}\,\|\,\mathsf{S}(B,x)\\ {[\blacktriangle_i/x]_i}\,B\end{array}\end{array}$$

- If $A = \forall_j y.\,B$, for $B \in \mathcal{F}$, then we can build:

$$\Phi = \mathsf{m}^{\exists}_{\forall}\,\frac{\exists_i x.\,\forall_j y.\,B}{\forall_j y\ \begin{array}{c}\exists_i x.\,B\\ \mathsf{IH}\,\|\,\mathsf{S}(B,x)\\ {[\blacktriangle_i/x]_i}\,B\end{array}}$$

- If $A = \langle t/y\rangle_j\, B$, for $t \in \ddot{\mathcal{T}}_j$ and $B \in \mathcal{F}$, then we can build:

$$\Phi = \sigma\exists\!\downarrow\,\frac{\exists_i x.\,\langle t/y\rangle_j\,B}{\left\langle \begin{array}{c}t\\ \Xi\,\|\,\mathsf{tw}\!\uparrow\\ {[\blacktriangle_i/x]_i}\,t\end{array}\Big/ y\right\rangle_j\ \begin{array}{c}\exists_i x.\,B\\ \mathsf{IH}\,\|\,\mathsf{S}(B,x)\\ {[\blacktriangle_i/x]_i}\,B\end{array}}$$

where Ξ is the derivation that coweakens on any occurrences of x appearing in t. If x does not appear free in t then $\Xi = t$ and there are no instances of $\mathsf{tw}\!\uparrow$, so Φ is a derivation in $\mathsf{S}(B,x)$.

◀

▶ **Corollary 48.** $\mathsf{v}\!\downarrow$ *is derivable in* KSqt *and* $\mathsf{v}\!\uparrow$ *is derivable in* SKSqt.

C.5 Identity

▶ **Lemma 49.** *For all* $t \in \ddot{\mathcal{T}}_i$*, there is a derivation* $\begin{array}{c}\top_0\\ \Phi\,\|\,\mathsf{KSqt}\\ t \mathrel{\mathring{=}_i} \bar{t}\end{array}$.

Proof. We proceed by induction on the structure of t.

- If $t \in \mathcal{C}_i \cup \mathcal{V}_i$ then $\Phi = \mathsf{tr}\!\downarrow\,\dfrac{\top_0}{t \mathrel{\mathring{=}_i} \bar{t}}$.

- If $t \in \{\triangledown_i, \blacktriangle_i\}$ then $\Phi = \mathsf{ur}\!\downarrow\,\dfrac{\top_0}{\triangledown_i \mathrel{\mathring{=}_i} \blacktriangle_i}$.

- If $t = s \mathbin{@_T} q$, for $s \in \ddot{\mathcal{T}}_{i+1}$ and $q \in \ddot{\mathcal{T}}_0$, then by induction we can build:

$$\Phi = \mathsf{ti}^{@}\!\downarrow\,\frac{\begin{array}{c}\top_0\\ \mathsf{IH}\,\|\\ s \mathrel{\mathring{=}_{i+1}} \bar{s}\end{array}\ \wedge_0\ \begin{array}{c}\top_0\\ \mathsf{IH}\,\|\,\mathsf{KSqt}\\ q \mathrel{\mathring{=}_0} \bar{q}\end{array}}{(s \mathbin{@_T} q) \mathrel{\mathring{=}_i} (\bar{s} \mathbin{@_T} \bar{q})}$$

- If $t = s \sqcup_i q$, for $s, q \in \dddot{\mathcal{T}}_i$, then by induction we can build:

$$\Phi = \mathsf{ti}^{\sqcup}\!\downarrow \frac{\begin{array}{c}\top_0\\ \mathsf{IH}\,\Vert\\ s \circeq_i \overline{s}\end{array} \;\wedge_0\; \begin{array}{c}\top_0\\ \mathsf{IH}\,\Vert\,\mathsf{KSqt}\\ q \circeq_i \overline{q}\end{array}}{(s \sqcup_i q) \circeq_i (\overline{s} \sqcap_i \overline{q})}$$

The case for $t = s \sqcap_i q$ is similar.

- If $t = \langle s/x \rangle_j\, q$, for $s \in \dddot{\mathcal{T}}_j$ and $q \in \dddot{\mathcal{T}}_i$, then by induction we can build:

$$\Phi = \mathsf{ti}^{\sigma}\!\downarrow \frac{\begin{array}{c}\top_0\\ \mathsf{IH}\,\Vert\\ s \circeq_j \overline{s}\end{array} \;\wedge_0\; \begin{array}{c}\top_0\\ \mathsf{IH}\,\Vert\,\mathsf{KSqt}\\ q \circeq_i \overline{q}\end{array}}{\langle s/x \rangle_j\, q \circeq_i \langle \overline{s}/x \rangle_j\, \overline{q}}$$

◀

▶ **Lemma 50.** *For all $A \in \dddot{\mathcal{A}}_i$, if $i > 0$ there exists a derivation* $\begin{array}{c}\top_0\\ \Phi\,\Vert\,\mathsf{KSqt}\\ A \leftrightharpoons_i \overline{A}\end{array}$ *and if $i = 0$ there exists a derivation* $\begin{array}{c}\top_0\\ \Phi\,\Vert\,\mathsf{KSqt}\\ A \vee_0 \overline{A}\end{array}$

Proof. We proceed by induction on the structure of A.

- If $A \in \mathcal{L}_i \cup \{\bot_i, \top_i\}$ then $\Phi = \mathsf{pr}\!\downarrow \dfrac{\top_0}{A \leftrightharpoons_i \overline{A}}$.
- If $A = B \mathbin{@_i} t$, for $B \in \dddot{\mathcal{A}}_{i+1}$ and $t \in \dddot{\mathcal{T}}_0$, then by induction and Lemma 49 we can build:

$$\Phi = \mathsf{pi}^{@}\!\downarrow \frac{\begin{array}{c}\top_0\\ \mathsf{IH}\,\Vert\,\mathsf{KSqt}\\ B \leftrightharpoons_{i+1} \overline{B}\end{array} \;\wedge_0\; \begin{array}{c}\top_0\\ \mathsf{Lem.49}\,\Vert\,\mathsf{KSqt}\\ t \circeq_0 \overline{t}\end{array}}{(B \mathbin{@_i} t) \leftrightharpoons_i (\overline{B} \mathbin{@_i} \overline{t})}$$

- If $A = B \vee_i C$, for $B, C \in \dddot{\mathcal{A}}_i$, then by induction we can build:

$$\Phi = \mathsf{pi}^{\vee}\!\downarrow \frac{\begin{array}{c}\top_0\\ \mathsf{IH}\,\Vert\,\mathsf{KSqt}\\ B \leftrightharpoons_i \overline{B}\end{array} \;\wedge_0\; \begin{array}{c}\top_0\\ \mathsf{IH}\,\Vert\,\mathsf{KSqt}\\ C \leftrightharpoons_i \overline{C}\end{array}}{(B \vee_i C) \leftrightharpoons_i (\overline{B} \wedge_i \overline{C})}$$

The case for $A = B \wedge_i C$ is similar.

- If $A = \langle t/x \rangle_j\, B$, for $t \in \dddot{\mathcal{T}}_j$ and $B \in \dddot{\mathcal{A}}_i$, then by induction and Lemma 49 we can build:

$$\Phi = \mathsf{pi}^{\sigma}\!\downarrow \frac{\begin{array}{c}\top_0\\ \mathsf{IH}\,\Vert\,\mathsf{KSqt}\\ B \leftrightharpoons_{i+1} \overline{B}\end{array} \;\wedge_0\; \begin{array}{c}\top_0\\ \mathsf{Lem.49}\,\Vert\,\mathsf{KSqt}\\ t \circeq_0 \overline{t}\end{array}}{\langle t/x \rangle_j\, B \leftrightharpoons_i \langle \overline{t}/x \rangle_j\, \overline{B}}$$

- Whenever we have $\begin{array}{c} \top_0 \\ \Phi \,\Vert\, \mathsf{KSqt} \\ A \leftrightharpoons_0 \bar{A} \end{array}$, we can append $\mathsf{i}^{\leftrightharpoons}_{\vee}\!\downarrow$ to obtain $\begin{array}{c} \top_0 \\ \Phi \,\Vert\, \mathsf{KSqt} \\ \mathsf{i}^{\leftrightharpoons}_{\vee}\!\downarrow \dfrac{A \leftrightharpoons_0 \bar{A}}{A \vee_0 \bar{A}} \end{array}$

- If $A = (B \leftrightharpoons_i C)$, for $B, C \in \mathcal{F}^{\mathcal{P}}_i$, then by induction we can build:

$$\Phi = \mathsf{i}^{\leftrightharpoons}\!\downarrow \dfrac{\begin{array}{c} \top_0 \\ \mathsf{Lem.50} \,\Vert\, \mathsf{KSqt} \\ B \leftrightharpoons_j \bar{B} \end{array} \wedge_0 \begin{array}{c} \top_0 \\ \mathsf{Lem.50} \,\Vert\, \mathsf{KSqt} \\ C \leftrightharpoons_i C^{\perp} \end{array}}{(B \leftrightharpoons_i C) \leftrightharpoons_0 (B^{\perp} \not\leftrightharpoons_i C^{\perp})}$$

The cases for $A = (B \not\leftrightharpoons_i C)$, $A = (s \mathrel{\mathring{=}}_i t)$, and $A = (s \not\mathrel{\mathring{=}}_i t)$ are similar, with the latter two following from Lemma 49. ◀

▶ **Corollary 51.** $\mathsf{ai}\!\downarrow$ *is derivable in* KSqt *and* $\mathsf{ai}\!\uparrow$ *is derivable in* SKSqt. ◀

C.6 Final Assembly for Completeness

Proof of Theorem 27. Every rule in $\mathsf{SKSq} \setminus \mathsf{SKSqt}$ is shown derivable in SKSqt by Corollaries 34, 37, 43, 48, and 51. ◀

D Proof of Conservativity

▶ **Definition 52.** *We define the following proof systems:*

- S_1 *consists of all inference rules in* $\mathsf{SKSqlx} \cup \{\mathsf{ai}'\!\downarrow, \mathsf{ai}'\!\uparrow\}$, *where* $\mathsf{ai}'\!\downarrow$ *and* $\mathsf{ai}'\!\uparrow$ *are obtained by extending* $\mathsf{ai}\!\downarrow$ *and* $\mathsf{ai}\!\uparrow$ *to the atoms of* SKSqt.
- S_2 *is obtained from* S_1 *by deleting all of the inference rules and equalities relating to* $\sqcup_i$, $\sqcap_i$ *for* $i \geq 0$ *and* $\vee_j$, $\wedge_j$ *for* $j > 0$ *and replacing them by* $\mathsf{ac}'\!\downarrow$ *and* $\mathsf{ac}'\!\uparrow$, *which are obtained by extending* $\mathsf{ac}\!\downarrow$ *and* $\mathsf{ac}\!\uparrow$ *to the atoms of* SKSqt. *More in detail, from the inference rules and equational theory given in Figure 2 we retain only* $\bot\triangledown\!\downarrow$, $\triangledown\triangledown\!\downarrow$, $\mathsf{pw}\!\downarrow$, $\mathsf{tw}\!\downarrow$ *and their duals. From the inference rules given in Figure 3 we delete* $\sigma\sqcap\!\downarrow$, $\sigma\wedge\!\downarrow$ *except for* $\wedge_0$, $\blacktriangle{*}\!\downarrow$ *for* $* \in \{\sqcup_i, \sqcap_i, \vee_j, \wedge_j \mid i \geq 0, j > 0\}$.
- S_3 *is obtained from* S_2 *by deleting all of the inference rules and equalities relating to* $\triangledown_i$, $\blacktriangle_i$ *for* $i \geq 0$, $\bot_j$, $\top_j$ *for* $j > 0$ *and replacing them by* $\mathsf{aw}'\!\downarrow$, $\mathsf{aw}'\!\uparrow$, $\mathsf{av}'\!\downarrow$ *and* $\mathsf{av}'\!\uparrow$, *which are obtained by extending* $\mathsf{aw}\!\downarrow$, $\mathsf{aw}\!\uparrow$, $\mathsf{av}\!\downarrow$ *and* $\mathsf{av}\!\uparrow$ *to the atoms of* SKSqt. *More in detail, we delete* $\bot\triangledown\!\downarrow$, $\triangledown\triangledown\!\downarrow$, $\mathsf{pw}\!\downarrow$, $\mathsf{tw}\!\downarrow$ *and their duals from Figure 2, and we delete* $\mathsf{qv}\!\downarrow$, $\blacktriangle\mathsf{v}\!\downarrow$, *all remaining cases of* $\blacktriangle{*}\!\downarrow$, $\blacktriangle\sigma\!\downarrow$, *and* $\blacktriangle\mathsf{Q}\!\downarrow$ *and their duals from Figure 3.*

▶ **Lemma 53.** *If* A, B *are* SKSq*-formulae and there is a derivation* $\begin{array}{c} A \\ \Phi \,\Vert\, \mathsf{SKSqt} \\ B \end{array}$ *then there is a derivation* $\begin{array}{c} A \\ \Xi_1 \,\Vert\, \mathsf{S}_1 \\ B \end{array}$. *Moreover,* Ξ_1 *does not contain any occurrences of the connectives* $\mathrel{\mathring{=}}_i$, $\not\mathrel{\mathring{=}}_i$, $\leftrightharpoons_i$, *or* $\not\leftrightharpoons_i$ *for* $i \geq 0$.

Proof. To prove this lemma we work in the extended proof system S_1^{+} consisting of all inference rules in $\mathsf{SKSqt} \cup \mathsf{S}_1$ together with all the inference rules in Figure 5 and their duals.

$$\mathsf{gtr}{\downarrow}\,\frac{\top_0}{t \circeq_i \bar{t}} \qquad \mathsf{gpr}{\downarrow}\,\frac{\top_0}{A \overleftarrow{=}_i \overline{A}} \qquad \mathsf{gw}^{\circeq}{\downarrow}\,\frac{\bot_0}{s \circeq_i t} \qquad \mathsf{gw}^{\overleftarrow{=}}{\downarrow}\,\frac{\bot_0}{A \overleftarrow{=}_i B} \qquad \mathsf{gw}^{\not\circeq}{\downarrow}\,\frac{\bot_0}{s \not\circeq_i t} \qquad \mathsf{gw}^{\not\overleftarrow{=}}{\downarrow}\,\frac{\bot_0}{A \not\overleftarrow{=}_i B}$$

$$\frac{s \circeq_i t}{(w \circeq_i s) \vee_0 (\overline{w} \not\circeq_i t)}\,(\mathsf{i}^{\circeq}{\downarrow}) \qquad \frac{(q \circeq_i w) \wedge_0 (r \circeq_i \overline{w})}{q \circeq_i r}\,(\mathsf{i}^{\circeq}{\downarrow}) \qquad \frac{A \overleftarrow{=}_i B}{(X \overleftarrow{=}_i A) \vee_0 (\overline{X} \not\overleftarrow{=}_i D)}\,(\mathsf{i}^{\overleftarrow{=}}{\downarrow}) \qquad \frac{(A \overleftarrow{=}_i X) \wedge_0 (B \overleftarrow{=}_i \overline{X})}{A \overleftarrow{=}_i B}\,(\mathsf{i}^{\overleftarrow{=}}{\downarrow})$$

$$\frac{s \circeq_i t}{\langle q/x\rangle_j\, s \circeq_i \langle \overline{q}/x\rangle_j\, t}\,(\mathsf{ti}^{\sigma}{\downarrow}) \qquad \frac{s \circeq_i t}{\langle s/x_i\rangle\, q \circeq_j \langle t/x_i\rangle\, \overline{q}}\,(\mathsf{ti}^{\sigma}{\downarrow}) \qquad \frac{s \circeq_0 t}{(q \mathbin{@_T} s) \circeq_i (\overline{q} \mathbin{@_T} t)}\,(\mathsf{ti}^{@}{\downarrow}) \qquad \frac{s \circeq_{i+1} t}{(s \mathbin{@_T} q) \circeq_i (t \mathbin{@_T} \overline{q})}\,\mathsf{ti}^{@}{\downarrow}$$

$$\frac{s \circeq_i t}{(s \sqcup_i q) \circeq_i (t \sqcap_i \overline{q})}\,(\mathsf{ti}^{\sqcup}{\downarrow}) \qquad \frac{s \circeq_0 t}{(A \mathbin{@_i} s) \overleftarrow{=}_i (\overline{A} \mathbin{@_i} t)}\,(\mathsf{pi}^{@}{\downarrow}) \qquad \frac{A \overleftarrow{=}_{i+1} B}{(A \mathbin{@_i} s) \overleftarrow{=}_i (B \mathbin{@_i} \overline{s})}\,(\mathsf{pi}^{@}{\downarrow}) \qquad \frac{s \circeq_i t}{\langle s/x_i\rangle\, A \overleftarrow{=}_j \langle t/x_i\rangle\, \overline{A}}\,(\mathsf{pi}^{\sigma}{\downarrow})$$

$$\frac{A \overleftarrow{=}_i B}{(A \vee_i X) \overleftarrow{=}_i (B \wedge_i \overline{X})}\,(\mathsf{pi}^{\vee}{\downarrow}) \qquad \frac{A \overleftarrow{=}_i B}{\langle s/x\rangle_j\, A \overleftarrow{=}_i \langle \overline{s}/x\rangle_j\, B}\,(\mathsf{pi}^{\sigma}{\downarrow})$$

Figure 5 The inference rules for the extended system S_1^+, used in the proof of Lemma 53. These inference rules are obtained by generalising inference rules in Figure 3 and they are either named as a general version of that rule or annotated on the right by the inference rule from which they are obtained.

These rules are all derivable in SKSqt and they will allow us to transform the derivation Φ into Ξ_1 in which identity is not local.

We do this by means of a rewriting relation that pushes the reflexivity rules $\mathsf{tr}{\downarrow}\,\dfrac{\top_0}{c \circeq_i c}$ and $\mathsf{pr}{\downarrow}\,\dfrac{\top_0}{P \overleftarrow{=}_i \overline{P}}$ through the proof until they meet the root $\mathsf{i}^{\overleftarrow{=}}_{\vee}{\downarrow}\,\dfrac{A \overleftarrow{=}_0 B}{A \vee_0 B}$ of the identity. We show some of the rewriting rules in Figure 6.

We can see the shape of this rewriting system by tracing occurrences of the connectives $\circeq_i$, $\not\circeq_i$, $\overleftarrow{=}_i$, or $\not\overleftarrow{=}_i$ through Φ, bifurcating the trace at rules such as $\mathsf{ti}^{@}{\downarrow}$ and $\mathsf{pi}^{\sigma}{\downarrow}$ and terminating at reflexivity rules or weakening rules such as $\mathsf{w}^{\overleftarrow{=}}{\downarrow}$. We know that A and B are SKSq-formulae, and therefore contain no occurrences of these connectives, so all of these traces terminate inside the proof in reflexivity or weakening rules.

In S_1^+ we can permute the general reflexivity and weakening rules through any other rules without creating more work to do, so this rewriting procedure will terminate, and in the normalised form the reflexivity rules will have accumulated into instances of $\mathsf{ai}'{\downarrow}$ and $\mathsf{ai}'{\uparrow}$. ◀

▶ **Lemma 54.** *If A, B are SKSq-formulae and there is a derivation* $\begin{array}{c} A \\ \Xi_1 \,\|\, \mathsf{S}_1 \\ B \end{array}$ *then there is a derivation* $\begin{array}{c} A \\ \Xi_2 \,\|\, \mathsf{S}_2 \\ B \end{array}$. *Moreover, Ξ_2 does not contain any occurrences of the connectives $\sqcup_i$, $\sqcap_i$, $\vee_j$, or $\wedge_j$, for $i \geq 0$ and $j > 0$.*

Proof. To prove this lemma we work in the extended proof system S_2^+ consisting of $\mathsf{S}_1 \cup \mathsf{S}_2$ together with all inference rules shown in Figure 7 and their duals. These additional inference

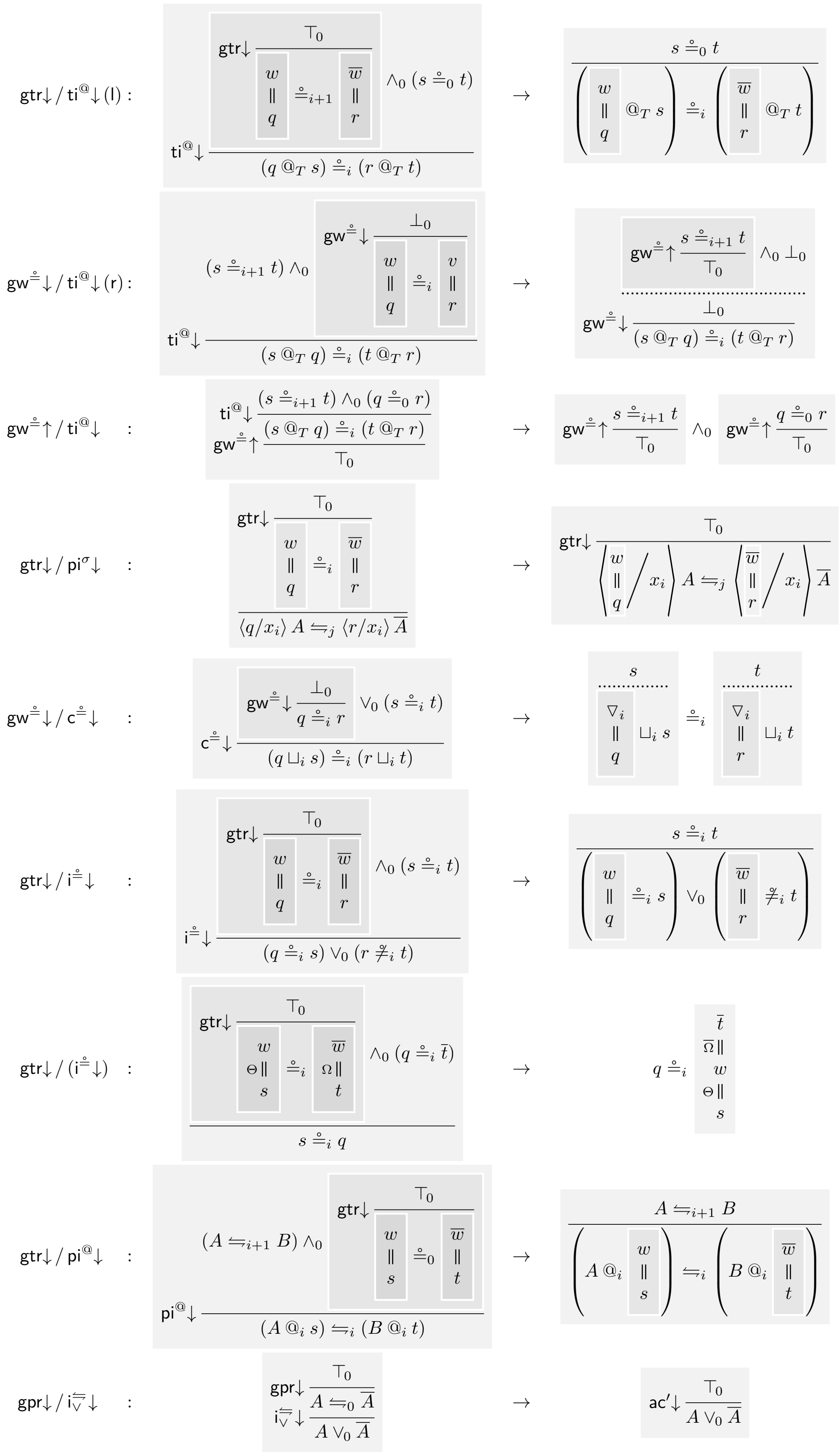


Figure 6 Some of the rewriting steps that we use in the proof of Lemma 53.

$$\mathsf{gtc}{\downarrow}\,\frac{t \sqcup_i t}{t} \qquad \mathsf{gpc}{\downarrow}\,\frac{A \vee_i A}{A} \qquad \top@{\downarrow}\,\frac{\top_i}{\top_{i+1} @_i s}$$

$$\mathsf{m}^{\sqcup}_{\sigma}\mathsf{l}\,\frac{\langle q/x\rangle_i\, s \sqcup_j \langle q/x\rangle_i\, t}{\langle q/x\rangle_i (s \sqcup_j t)} \qquad \mathsf{m}^{\sqcup}_{\sigma}\mathsf{r}\,\frac{\langle s/x_i\rangle\, q \sqcup_j \langle t/x\rangle_i\, q}{\langle s \sqcup_i t/x\rangle_i\, q} \qquad \mathsf{m}^{\vee}_{\sigma}\mathsf{l}\,\frac{\langle q/x_i\rangle\, A \vee_j \langle q/x\rangle_i\, B}{\langle q/x\rangle_i (A \vee_j B)} \qquad \mathsf{m}^{\vee}_{\sigma}\mathsf{r}\,\frac{\langle s/x\rangle_i\, A \vee_j \langle q/x\rangle_i\, A}{\langle s \sqcup_i q/x\rangle_i}$$

$$\mathsf{m}^{\sqcup}_{@}\mathsf{l}\,\frac{(q @_T s) \sqcup_i (q @_T t)}{q @_T (s \sqcup_0 t)} \qquad \mathsf{m}^{\sqcup}_{@}\mathsf{r}\,\frac{(s @_T q) \sqcup_i (t @_T q)}{(s \sqcup_i t) @_T q} \qquad \mathsf{m}^{\vee}_{@}\mathsf{l}\,\frac{(A @_i s) \vee_i (A @_i q)}{A @_i (s \sqcup_0 q)} \qquad \mathsf{m}^{\vee}_{@}\mathsf{r}\,\frac{(A @_i q) \vee_i (B @_i q)}{(A \vee_i B) @_i q}$$

$$\mathsf{m}^{\sqcup}_{\sqcap}\mathsf{c}\,\frac{(s \sqcap_i q) \sqcup_i (t \sqcap_i q)}{(s \sqcup_i t) \sqcap_i q} \qquad \mathsf{mc}\,\frac{(A \wedge_i B) \vee_i (A \wedge_i C)}{A \wedge_i (B \vee_i C)}$$

Figure 7 The inference rules for the extended system S_2^+, used in the proof of Lemma 54

rules are all derivable in S_1 and they will allow us to transform the derivation Ξ_1 into Ξ_2 in which contraction is not local.

We begin by noting that

$$\frac{(A @_i s) \vee_i \top_i}{(A \vee_{i+1} \top_{i+1}) @_i s}$$ is derivable S_2^+ by means of the derivation

$$\mathsf{m}^{\vee}_{@}\mathsf{r}\,\dfrac{(A @_i s) \vee_i \;\; \top@{\downarrow}\,\dfrac{\top_i}{\top_{i+1} @_i s}}{(A \vee_i \top_i) @_i s}$$

so we can eliminate any instances of that rule.

We then proceed in a similar fashion as Lemma 53, permuting instances of generalised term and predicate (co)contraction through the derivation until they accumulate into atomic (co) contractions and permuting equality rules until they annihilate. We give a sample of the rewriting rules in Figure 8. These rewriting steps do not increase the arity of what is being contracted, so it will terminate, resulting in an S_2 derivation without any occurrences of those connectives. ◀

▶ **Lemma 55.** *If A, B are* SKSq*-formulae and there is a derivation* $\begin{array}{c} A \\ \Xi_2 \,\| \,\mathsf{S}_2 \\ B \end{array}$ *then there is a derivation* $\begin{array}{c} A \\ \Xi_3 \,\| \,\mathsf{S}_3 \\ B \end{array}$. *Moreover, Ξ_3 does not contain any occurrences of the units $\triangledown_i$, $\blacktriangle_i$, $\bot_j$, or $\top_j$, for $i \geq 0$ and $j > 0$.*

Proof. To prove this lemma we extend S_3 by the inference rules in Figure 9 and their duals to obtain the proof system S_3^+. These inference rules are all derivable in S_2 and they will allow us to transform the derivation Ξ_2 into Ξ_3 in which weakening and vacuous quantification are not local.

We will consider the procedure for recovering $\mathsf{aw}{\downarrow}$ and $\mathsf{av}{\uparrow}$ because these both concern the unit $\triangledown_i$ and so this illustrates the rewriting system more clearly than considering the down rules together. Some of the rewriting rules are given in Figure 10.

We deal with vacuous quantification by first transforming prefixed derivations into standard derivations by de-parallelising them. Then we permute the application of explicit

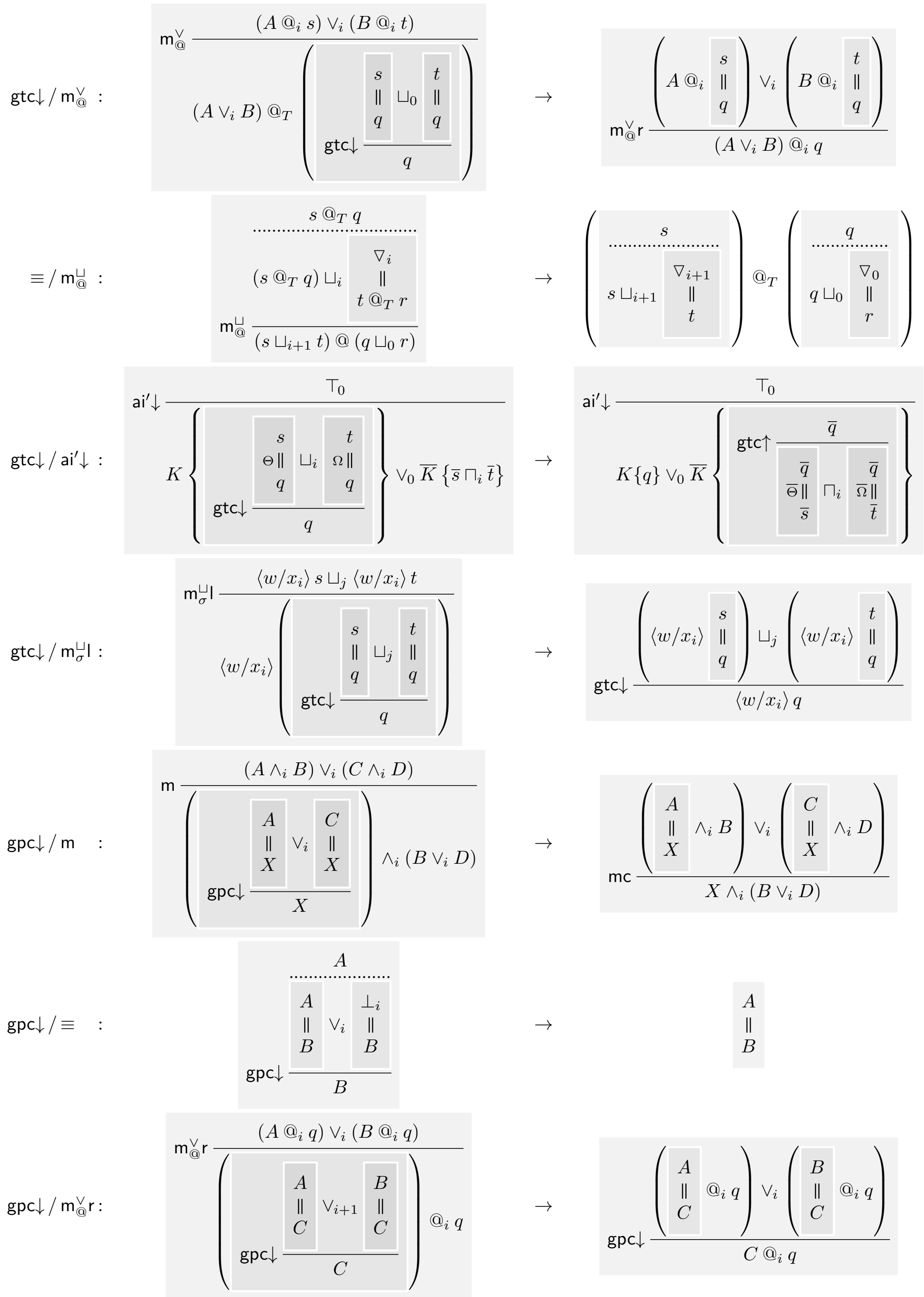


Figure 8 Some of the rewriting steps that we use in the proof of Lemma 54.

$$\mathsf{twl}\!\downarrow \frac{\triangledown_i}{s \mathbin{@_T} \triangledown_0} \qquad \mathsf{twr}\!\downarrow \frac{\triangledown_i}{\triangledown_{i+1} \mathbin{@_T} t} \qquad \mathsf{pwl}\!\downarrow \frac{\bot_i}{A \mathbin{@} \triangledown_0} \qquad \mathsf{pwr}\!\downarrow \frac{\bot_i}{\bot_{i+1} \mathbin{@_i} t} \qquad \mathsf{gtw}\!\downarrow \frac{\triangledown_i}{t} \qquad \mathsf{gpw}\!\downarrow \frac{\bot_i}{A}$$

$$\triangledown{*}{\uparrow}\mathsf{r} \frac{\langle \triangledown_i/x\rangle_i A * [\triangledown_i/x]_i B}{\langle \triangledown_i/x\rangle_i (A * B)} \qquad \triangledown{*}{\uparrow}\mathsf{l} \frac{[\triangledown_i/x]_i A * \langle \triangledown_i/x_i\rangle B}{\langle \triangledown_i/x\rangle_i (A * B)} \qquad \frac{[\triangledown_i/x]_i A}{\langle \triangledown_i/x\rangle_i A} \qquad \frac{\langle \triangledown_i/x\rangle_i A}{\forall_i x A} \qquad \frac{[\triangledown_i/x]_i A}{\forall_i x A}$$

$$\triangledown{*}{\uparrow}\mathsf{l} \frac{\langle [\triangledown_i/x]_i t/y\rangle_j \langle \triangledown_i/x\rangle_i A}{\langle \triangledown_i/x\rangle_i \langle t/y\rangle_j A} \qquad \triangledown\sigma{\uparrow}\mathsf{r} \frac{\langle \langle \triangledown_i/x\rangle_i t/y\rangle_j [\triangledown_i/x]_i A}{\langle \triangledown_i/x\rangle_i \langle t/y\rangle_j A}$$

Figure 9 The generalised inference rules for eliminating $\triangledown_i$. Note that because $\mathsf{v}\!\uparrow$ and $\mathsf{w}\!\downarrow$ both concern $\triangledown_i$, the rules here for explicit substitutions are from the up-fragment, the generalised inference rules for eliminating $\mathsf{v}\!\downarrow$ and $\mathsf{w}\!\uparrow$ being dual.

$$\triangledown\triangledown\!\downarrow \frac{\triangledown_i}{\left(\mathsf{gtw}\!\downarrow \frac{\triangledown_{i+1}}{s}\right) \mathbin{@_T} t} \quad \to \quad \mathsf{gtw}\!\downarrow \frac{\triangledown_i}{s \mathbin{@_T} t}$$

$$\bot\triangledown\!\downarrow \frac{\bot_i}{\left(\mathsf{gpw}\!\downarrow \frac{\bot_{i+1}}{A}\right) \mathbin{@_T} t} \quad \to \quad \mathsf{gpw}\!\downarrow \frac{\bot_i}{A \mathbin{@_i} t}$$

$$\left(\mathsf{qv}\!\uparrow \frac{\langle \triangledown_i/x_i\rangle}{\forall_i x}\right) \left(\begin{array}{c} A \\ \| \\ B \end{array}\right) \quad \to \quad \frac{\langle \triangledown_i/x\rangle_i A}{\forall_i x \left(\begin{array}{c} A \\ \| \\ B \end{array}\right)}$$

$$\triangledown@\!\uparrow \frac{\frac{[\triangledown_i/x]_i A}{\langle \triangledown_i/x\rangle_i A} \mathbin{@_i} \langle \triangledown_i/x\rangle_i t}{\langle \triangledown_i/x\rangle_i (A \mathbin{@_i} t)} \quad \to \quad \triangledown@\!\uparrow\!\mathsf{l} \frac{[\triangledown_i/x]_i A \mathbin{@_i} \langle \triangledown_i/x_i\rangle t}{\langle \triangledown_i/x\rangle_i (A \mathbin{@_i} t)}$$

$$\frac{\frac{[\triangledown_i/x]_i B \mathbin{@_j} \langle \triangledown_i/x\rangle_i t}{\langle \triangledown_i/x\rangle_i (B \mathbin{@_j} t)}}{\forall_i x (B \mathbin{@_j} t)} \quad \to \quad \frac{[\triangledown_i/x]_i B \mathbin{@_j} \langle \triangledown_i/x\rangle_i t}{\forall_i x (B \mathbin{@_j} t)}$$

Figure 10 Some of the rewriting steps used in the proof of Lemma 55.

substitutions towards instances of $\mathsf{qv}\!\uparrow \frac{\langle \triangledown_i/x_i\rangle}{\forall_i x}$, transforming these into $\frac{[\triangledown_i/x]_i A}{\forall_i x A}$. We can also permute the term and predicate weakening rules through the proof so that they accumulate into atomic weakening rules. Because we know that these units do not appear in A or B, we know that they must terminate inside Ξ_2 at a weakening, vacuous application, or vacuous quantification, so we can permute these inference rules until no occurrences of these units remain. ◀

▶ **Lemma 56.** *If A, B are* SKSq*-formulae and there is a derivation* $\begin{array}{c} A \\ \Xi_3 \, \| \, \mathsf{S}_3 \\ B \end{array}$ *then there is*

$$\dfrac{[s/x]_i\, A}{\langle s/x\rangle_i\, A} \qquad \dfrac{\langle t/x\rangle_i\, A}{\exists_i x A} \qquad \dfrac{\exists_{i+1} y\, [y \mathbin{@_T} t/x]_i\, A}{\exists_i x A} \qquad \dfrac{\exists_0 z\, [s \mathbin{@_T} z/x]_i}{\exists_i x A}$$

Figure 11 The additional inference rules used in the proof of Lemma 56.

$$\left(\begin{array}{l} \mathsf{qn}\!\uparrow \dfrac{\forall_i x}{\langle t/x_i\rangle} \\ \mathsf{qn}\!\downarrow \dfrac{}{\exists_i x} \end{array} \right) A \quad \to \quad \begin{array}{c} \forall_i x A \\ \hline \langle t/x_i\rangle\, A \\ \hline \exists_i x A \end{array} \quad \to \quad \begin{array}{rc} \mathsf{n}\!\uparrow & \forall_i x A \\ \cline{2-2} & [t/x_i]\, A \\ \cline{2-2} \mathsf{n}\!\downarrow & \exists_i x A \end{array}$$

$$\begin{array}{rc} & \exists_0 z\, [f \mathbin{@_T} z/x_i]\, A \\ \cline{2-2} & \exists_{i+1} y \exists_0 z\, [y_{i+1} \mathbin{@_T} z_0/x_i]\, A \\ \cline{2-2} \mathsf{x}\!\downarrow & \exists_i x A \end{array} \quad \to \quad \dfrac{\exists_0 z\, [f \mathbin{@_T} z/x_i]\, A}{\exists_i x A}$$

Figure 12 Some of the rewriting steps that we use in the proof of Lemma 56

a derivation $\begin{array}{c} A \\ \Xi_4 \, \Vert\, \mathsf{SKSq} \\ B \end{array}$ *. Moreover,* Ξ_4 *does not contain any instances of composition by explicit substitution, prefix derivations, or variables* x_i *for* $i > 0$.

Proof. To prove this lemma we extend SKSq by the inference rules in Figure 11 and their duals to obtain the proof system S_4^+. These inference rules are all derivable in S_3 and they will allow us to transform the derivation Ξ_3 into Ξ_4 in which instantiation is not local.

We note that for all terms t appearing in Ξ_3, $\bar{t} = t$, so we can apply explicit substitutions freely without concern for duality in the substituted term. It only remains to de-parallelise any prefix derivations and permute the application of explicit substitutions towards the instantiation rules that created them.

Because A and B are SKSq-formulae, they cannot contain any variables x_i for $i > 0$, so this rewriting procedure eliminates any such variables from the derivation.

One inference rule from S_4^+ is immune to this rewriting procedure: $\dfrac{\exists_0 z\, [f \mathbin{@_T} z/x_0]\, A}{\exists_0 x A}$.

This rule doesn't contain any higher arity variables and so doesn't get eliminated in the case where z_0 is never instantiated. This rule is derivable in SKSq, so we replace it by its derivation:

$$\mathsf{s}\,\dfrac{\exists_0 z\, [f \mathbin{@_T} z/x_0]\, A \wedge \left(\exists_0 x A \vee \left(\begin{array}{c} \top \\ \Vert\, \mathsf{ai}\!\downarrow,\mathsf{s},\mathsf{qs}\!\downarrow \\ \begin{array}{c} \forall_0 x \overline{A} \\ \Vert\, \mathsf{av}\!\uparrow,\mathsf{m} \\ \forall_0 z \left(\mathsf{n}\!\uparrow \dfrac{\forall_0 x \overline{A}}{[f \mathbin{@_T} z/x]_0\, \overline{A}} \right) \end{array} \end{array} \right) \right)}{\left(\begin{array}{c} \exists_0 z\, [f \mathbin{@_T} z/x]_0\, A \wedge \forall_0 z\, [f \mathbin{@_T} z/x]_0\, \overline{A} \\ \Vert\, \mathsf{ai}\!\uparrow,\mathsf{s},\mathsf{qs}\!\uparrow \\ \bot \end{array} \right) \vee \exists_0 x A}$$

The result is a derivation in SKSq. ◀

E Variable Naming Conventions and Freshness

In the inference systems depicted in Figures 2–4, there are several instances of side condition requiring a certain binding occurrence of a variable to be a "fresh" variable. Naïvely, this can be seen as a global check on the entire derivation, similar to the non-local vacuousness check in av↓ and av↑ in SKSq (Figure 1). In this paper, we avoid this global check by insisting that derivations adhere to the following variable naming convention: every introduction of a binder $\exists_i x$, $\forall_i x$, and $\langle t/x \rangle_i$ in an inference rule (read in either direction) must bind a globally unique identifier x, i.e., x is not bound by a binder introduced in any other rule in the derivation. We additionally require that the binders in the premise and conclusion of a derivation bind pairwise distinct variables. These naming conventions can be seen in the same spirit as the requirement that the applications of compound terms and atoms are arity-correct.

We summarise here how these naming conventions can be implemented with essentially constant cost operations. The core idea is, unsurprisingly, to use a syntax using De Bruijn indices, so that bound variables are no longer named and α-conversion has zero cost. However, there are a few complications. First, consider a rule such as the following:

$$\frac{\exists_j y.\, \exists_i x.\, A}{\exists_i x.\, \exists_j y.\, A}$$

Because the order of variables changes, the body A needs to adjusted to swap the indices for x and y. One way to do this locally is to introduce a new *permuting substitution* term constructor $\pi(x, y)$, which has the same status as an explicit subsitution—i.e., it is part of the quantifier prefix—but it binds no variables; instead, it merely symmetrically permutes (the indices of) x and y. The rule can then be written as follows:

$$\frac{\exists_j x.\, \exists_i y.\, \pi(x, y).\, A}{\exists_i x.\, \exists_j y.\, A}$$

This permutation has the obvious analogue of the explicit substitution rules shown in Figure 3, which incrementally moves it to the leaves of A along with other elementary substitutions.

A more interesting issue arises for a rule such as x↓ which increases the length of the quantifier prefix.

$$\mathsf{x}\!\downarrow \frac{\exists_{i+1} y.\, \exists_0 z.\, \langle y \mathbin{@_T} z/x \rangle_i}{\exists_i x}$$

To implement this rule locally, we need to add an additional formal variable renumbering substitution, $\Uparrow_k^n$, which leaves indices $0, \ldots, k-1$ alone, and raises the indices $k, k+1, \ldots$ to $k+n, k+n+1, \ldots$. This substitution also has the obvious distributivity rules similar to that of explicit substitutions in Figure 3. The local form of the x↓ rule then becomes:

$$\mathsf{x}\!\downarrow \frac{\exists_{i+1} y.\, \exists_0 z.\, \langle y \mathbin{@_T} z/x \rangle_i \,.\, \Uparrow_0^2}{\exists_i x}$$

These implementation details of the variable convention are important for a fully detailed view of locality, but detract from the core contribution of this paper, so we keep the presentation here in its simpler named form.