Report Number: WUCSE-2008-7

2008-01-01

Deciding Joinability Modulo Ground Equations In Operational Type Theory

Adam Petcher and Aaron Stump

Operational Type Theory (OpTT) can be used to construct and check proofs related to programs, but the development of these proofs can be somewhat tedious. An algorithm is presented that can be used to automatically generate proofs of equality in OpTT. The algorithm takes as input a set of ground equations and two terms that should be tested for joinability modulo the supplied ground equations. The algorithm will equate the terms if and only if there exists an OpTT proof that can equate the two terms using only the proof rules related to evaluation under the operational semantics, symmetry,... **Read complete abstract on page 2.**

Deciding Joinability Modulo Ground Equations In Operational Type Theory

Adam Petcher and Aaron Stump

Complete Abstract:

Operational Type Theory (OpTT) can be used to construct and check proofs related to programs, but the development of these proofs can be somewhat tedious. An algorithm is presented that can be used to automatically generate proofs of equality in OpTT. The algorithm takes as input a set of ground equations and two terms that should be tested for joinability modulo the supplied ground equations. The algorithm will equate the terms if and only if there exists an OpTT proof that can equate the two terms using only the proof rules related to evaluation under the operational semantics, symmetry, transitivity, and congruence with respect to the supplied ground equations. The description of this algorithm is accompanied by a proof that the algorithm is partially correct.

2008-7

Deciding Joinability Modulo Ground Equations In Operational Type Theory

Authors: Adam Petcher, Aaron Stump

Abstract: Operational Type Theory (OpTT) can be used to construct and check proofs related to programs, but the development of these proofs can be somewhat tedious. An algorithm is presented that can be used to automatically generate proofs of equality in OpTT. The algorithm takes as input a set of ground equations and two terms that should be tested for joinability modulo the supplied ground equations. The algorithm will equate the terms if and only if there exists an OpTT proof that can equate the two terms using only the proof rules related to evaluation under the operational semantics, symmetry, transitivity, and congruence with respect to the supplied ground equations. The description of this algorithm is accompanied by a proof that the algorithm is partially correct.

Type of Report: Other

WASHINGTON UNIVERSITY
SCHOOL OF ENGINEERING AND APPLIED SCIENCE
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECIDING JOINABILITY MODULO GROUND EQUATIONS IN OPERATIONAL TYPE THEORY

by

Adam Petcher

Prepared under the direction of Professor Aaron Stump

A thesis presented to the School of Engineering of
Washington University in partial fulfillment of the
requirements for the degree of
MASTER OF SCIENCE

May 2008
Saint Louis, Missouri

WASHINGTON UNIVERSITY
SCHOOL OF ENGINEERING AND APPLIED SCIENCE
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ABSTRACT

DECIDING JOINABILITY MODULO GROUND EQUATIONS IN OPERATIONAL TYPE THEORY

by

Adam Petcher

ADVISOR: Professor Aaron Stump

May 2008
St. Louis, Missouri

Operational Type Theory (OpTT) can be used to construct and check proofs related to programs, but the development of these proofs can be somewhat tedious. An algorithm is presented that can be used to automatically generate proofs of equality in OpTT. The algorithm takes as input a set of ground equations and two terms that should be tested for joinability modulo the supplied ground equations. The algorithm will equate the terms if and only if there exists an OpTT proof that can equate the two terms using only the proof rules related to evaluation under the operational semantics, symmetry, transitivity, and congruence with respect to the supplied ground equations. The description of this algorithm is accompanied by a proof that the algorithm is partially correct.

Table of Contents

List of Figures

1 Introduction

Operational Type Theory(OpTT)[8] is a powerful and convenient system for developing programs and proofs related to those programs. In some cases, however, the development of proofs in OpTT can be excessively tedious. This paper describes a class of proofs that could be automatically generated by an algorithm that is capable of deciding the "joinability modulo ground equations" problem. Such an algorithm would significantly reduce the amount of effort required to develop proofs in OpTT.

The primary contribution of this paper is an algorithm which attempts to decide the joinability modulo ground equations problem. Also included is a proof that the algorithm is correct when it terminates and certain "consistency" conditions are satisfied. An analysis of the termination properties of the algorithm is not included, but it is conjectured that the algorithm will terminate if recursion is finite in the supplied program terms and in all terms that are derived by the algorithm.

This chapter describes OpTT and introduces the joinability modulo ground equations problem.

1.1 Operational Type Theory

Operational Type Theory is a system in which programs and proofs have distinct type systems, but can nevertheless be combined such that a program can contain a proof, and a proof related to a program can be documented and checked. The primary motivation for this distinction is to allow for the independence of the semantics of program evaluation, proof normalization, and definitional equality. As a result of this independence, proof complexity is not dependent on program complexity, and it is possible to develop simple proofs related to complex programs.

1.2 Proof Complexity in OpTT

In many practical cases, proofs in OpTT tend to be mostly composed of tedious sub-proofs that could easily be generated using simple automated reasoning. These proofs are usually composed of instances of a handful of common patterns, and they tend to have relatively large amounts of redundant information. If the need to manually develop such trivial proofs was removed, then proof development in OpTT would be significantly less time consuming.

A simple example will be used to further describe the problem. Additional examples are used to provide more detail in chapter 2. Several examples in this paper assume the definitions in figure 1-1 which define natural numbers, Booleans, and ≥ for natural numbers, respectively.

```
Inductive nat : type :=
  Z : nat
| S : Fun(x:nat).nat.

Inductive bool : type :=
  T : bool
| F : bool.

Define ge : Fun(n m : nat). bool :=
    fun (n m : nat):bool.
    match m by x y return bool with
      Z => T
    | S m' => match n by x1 y1 return bool with
        Z => F
      | S n' => (ge n' m')
      end
    end.
```

Figure 1-1. Example Definitions

One might prove that ge is reflexive, that is, for all n, (ge n n) = T, as follows. Proof by induction on n: In the base case, n = Z. Because n=Z and congruence of terms w.r.t. equality of subterms, (ge n n) = (ge Z Z). Due to the definition of ge, (ge Z Z) = T. From transitivity of equality, (ge n n) = T. In the step case, n = (S n') and there is an induction hypothesis that says (ge n' n') = T. Because n = (S n') and congruence of terms, (ge n n) = (ge (S n') (S n')). Due to the definition of ge, (ge (S n') (S n')) = (ge n' n'). Due to the induction hypothesis and transitivity of equality, (ge n n) = T. This proof in OpTT syntax is shown in figure 1-2. Note that trans, cong, and join are the proof rules for transitivity of equality, congruence of terms, and evaluation under the operational semantics, respectively.

```
Define geRefl : Forall(n:nat). { (ge n n) = T } :=
    induction(n:nat) by x1 x2 IH return { (ge n n) = T} with
        Z => trans
                cong (ge * *) x1
                join (ge Z Z) T
    | S n' => trans
                trans
                    cong (ge * *) x1
                    join (ge (S n') (S n')) (ge n' n')
                [IH n']
    end.
```

Figure 1-2. Standard Proof of Reflexivity of ge

Each case of the induction is little more than a series of cong and join rules linked together by trans rules. It is common for proofs in OpTT to be composed of such large sub-proofs that contain only cong, join, symm, and trans rules. The cong proof rule also includes a ground equation as an argument, so this class of OpTT proof will be referred to as a joinability modulo ground equations proof, and the problem of determining the existence of such proofs is called the joinability modulo ground equations problem. Because the need for this sort of proof is so common, an algorithm which can decide the joinability modulo

ground equations problem would greatly simplify proofs in many cases. For example, it would be desirable to simply write the proof from figure 1-2 as shown in figure 1-3.

```
Define geRefl : Forall(n:nat). { (ge n n) = T } :=
        induction(n:nat) by x1 x2 IH return { (ge n n) = T} with
              Z => hypjoin (ge n n) T by x1 end
            | S n' => hypjoin (ge n n) T by x1 [IH n'] end
        end.
```

Figure 1-3. Proof of Reflexivity of ge using hypjoin

Note that hypjoin is the name given to the new proof rule in OpTT that can algorithmically decide the joinability modulo ground equations problem. The "hyp" in hypjoin stands for hypothesis, and the name "hypjoin" represents the fact that we are attempting to join two terms given a set of user-provided hypotheses. This problem is described in greater detail in chapter 2.

Though the example in figure 1-2 was very simple, the purpose of hypjoin is to take large, complicated proofs and collapse them into a single hypjoin statement. More complex examples are provided in section 2.5. The goal of this paper is to define an algorithm that implements hypjoin and to prove that the algorithm is correct to the extent that it can be used in practical cases. The algorithm is defined in chapter 3, and the proof of correctness is provided in chapter 4.

1.3 Related Work

Congruence closure[2] is a decision procedure that can be used to decide the equivalence of two terms given a finite set of ground equations E that defines a set of allowed substitutions. This problem is known as the ground word problem for E[1]. Congruence closure is a significant part of the hypjoin algorithm, but using it to solve the entire problem is

challenging because it is difficult to effectively represent the operational semantics as a finite set of ground equations.

The Calculus of Congruent Inductive Constructions(CCIC)[3][4] is an extension to the Calculus of Inductive Constructions(CIC) that incorporates decision procedures for first-order theories into the CIC conversion rule. This system is very powerful in that the user-supplied hypotheses can be equations containing quantified variables, and as such it can simplify many types of complex proofs. Though hypjoin is less general than CCIC, the fact that it is limited to reasoning on ground equations provides some significant benefits when compared to CCIC. The primary benefit is that the computational nature of the hypjoin algorithm will probably make it more understandable to the programmer. If hypjoin fails, an error message will be presented that can help the programmer understand why the failure occurred. If conversion fails, however, it may not be possible to generate such an error message. Another benefit of hypjoin is that it can be seen as an optional extension to OpTT, whereas the modified conversion rule is a core part CCIC. Due to this structure, the theoretical properties of OpTT (cut elimination, type soundness) are not affected by hypjoin.

2 Problem Description

This section contains a discussion of the issues surrounding the joinability modulo ground equations problem. Section 2.1 contains a definition of the problem, and a description of the difficulties inherent to solving the problem is presented in section 2.2. Section 2.3 presents a set of achievable goals that would allow an algorithm that partially decides the joinability modulo ground equations problem to be useful in practical cases.

2.1 Problem Definition

The joinability modulo ground equations problem can be described informally as follows. Given two terms, s and t, and a set of ground equations U, is there a path from s to t using evaluation under the operational semantics and substitutions from U? This paper is concerned with solving this problem using the operational semantics and constraints of OpTT. The operational semantics of OpTT can be defined as follows.

The definitions of contexts E and E^+ are found in [8] and reproduced in figure 2-1 for convenience. The definition of E^+ excludes contexts that are types since it is assumed that all type annotations have been dropped in the terms and equations that are provided to hypjoin. It is often helpful to consider contexts that are exactly one level deep. These contexts are described by the symbols E_1 and E_1^+ which are identical to E and E^+, respectively, except the first set of contexts are one level deep. For example, $E_1^+[a]$ could describe (a b), but not ((a b) c). Another difference is that the one-level-deep contexts could have an empty set of holes, as opposed to traditional contexts that must have at least one hole, but the context itself could be empty. The context E_1^* is identical to E_1^+ except it has an arbitrary number of (zero or more) holes. Notation such as $E_1^*[\overline{a}]$ refers to a context and the list of terms $(\overline{a})$, in the holes of that context. Any time this notation is used, there is an element in $\overline{a}$ for every possible hole in E_1^* based on the form of the term. For example, if $E_1^*[\overline{a}]$ refers to a let term, then there must be exactly two elements in $\overline{a}$.

E ::= * | | (E X) | | (I E) | |
let x = E by y in y | |
match E by x y return T with
$c_1\overline{x}_1 => t_1 | ... | c_n\overline{x}_n => t_n$ end

E^+ ::= * | | X | | $(E^+\ E^+)$ | |
fun r($\overline{x} : \overline{E}^+$) : $E^+ . E^+$ | |
let x = E^+ by y in E^+ | |
match E^+ by x y return E^+ with
$c_1\overline{x}_1 => E^+ | ... | c_n\overline{x}_n => E^+$ end

Figure 2-1. Definitions of Contexts

When using one-level-deep contexts, one can easily extract the bound variables defined in that context. Later definitions will refer to a function called Bound that is used to produce a set of bound variables for a given one-level deep context. Bound is defined in figure 2-2.

Bound((a1 a2), 1) = ∅
Bound((a1 a2), 2) = ∅
Bound(fun r(x:!):!.a, 1) = {x}
Bound(let x = a1 in a2, 1) = ∅
Bound(let x = a1 in a2, 2) = {x}
Bound(match s return ! with $c_1\overline{x}_1 => a_1 | \ldots | c_n\overline{x}_n => a_n | \ldots$end, 1) = ∅
Bound(match s return ! with $c_1\overline{x}_1 => a_1 | \ldots | c_n\overline{x}_n => a_n | \ldots$end, m) = $\{\overline{x}_{m-1}\}$

Figure 2-2. Definition of Bound

For any context E with only one hole, Bound(E) is shorthand for Bound(E, k) where k is the position of the only hole in E. We can also apply Bound to an evaluation context with only one hole but arbitrary depth. In which case the function is defined recursively as shown in figure 2-3.

$\text{Bound}(*) = \varnothing$

$\text{Bound}(E + [E_1^+]) = \text{Bound}(\text{E+}) \cup \text{Bound}(E_1^+)$

Figure 2-3. Recursive Definition of Bound

The definition of the evaluation relation is provided in figures 2-4, 2-5, and 2-6. The definition provided uses the one level deep context defined earlier, but is nevertheless equivalent to the operational semantics defined in [8]. Note that $\rightarrow$ is the symbol used to represent a single step of evaluation.

The relation is defined using a set of rules which combine to form an inductive definition. That is, any time two terms are related by $\rightarrow$, there must exist a derivation composed of these rules which relate the terms. The notation k:aRb is used to represent the fact that a is related to b in inductively-defined relation R by a derivation of depth k. This convention is used any time a relation is defined inductively in this paper.

$$\frac{f = fun\ r(x:!):!.b}{(f\ I) \rightarrow b[I/x][f/r]}$$

$$\overline{match(c_i\ \overline{I})\ return\ !\ with\ c_1\overline{x}_1 => s_1 \mid ... \mid c_n\overline{x}_n => s_n\ end \quad \rightarrow \quad s_i[\overline{I}/\overline{x}_i]}$$

$$\overline{let\ x = I\ in\ t \quad \rightarrow \quad t[I/x]}$$

$$\overline{E_1[abort\ !] \quad \rightarrow \quad abort\ !}$$

$$\frac{t \rightarrow t'}{E_1[t] \quad \rightarrow \quad E_1[t']}$$

Figure 2-4. Inductive Definition of Operational Semantics of OpTT

Note that Γ is a set of bound variables and, in general, $\Gamma \triangleright aRb$ means that a and b are related in R and no bound variables, as defined by Γ, are introduced or eliminated when a is replaced with b.

$\dfrac{\Gamma \cap (Vars(s_n) \cup Vars(t_n)) = \varnothing}{\Gamma \rhd t_n[\overline{s} / \overline{t}] = s_n}$	CAS-BASE
$\dfrac{\forall n \quad \Gamma, Bound(E_1^*, n) \rhd a_n[\overline{t} / \overline{s}] = b_n}{\Gamma \rhd (E_1^*[\overline{a}])[\overline{t} / \overline{s}] = E_1^*[\overline{b}]}$	CAS-IND

Figure 2-5. Inductive Definition of Capture-Avoiding Substitution

$$\frac{(Vars(s) \cup Vars(t)) \cap \Gamma = \varnothing \quad s \to t}{\Gamma \rhd s \to t}$$

Figure 2-6. Bound Variable-Preserving Evaluation

The set of ground equations will be referred to as U and has the following form: $\{u_1 = u_1', u_2 = u_2', ..., u_n = u_n'\}$. Two terms, s and t, are equal modulo a set of ground equations U if there exists a sequence of substitutions defined by U that can be applied to s and t to produce two definitionally equal terms. This equality is defined in figure 2-7.

It is assumed that the variables in U only refer to free variables in s and t. If there happens to be a bound variable in s or t with the same name as a variable in U, then these should be treated as different variables. Techniques such as de Bruijn indices[5] can be utilized to prevent inadvertent variable name collisions. As a result of this assumption, equality modulo U is not constrained by any context.

Note that in the EQ-REFL case, two terms are considered to be the same term if they are definitionally equal. It should be apparent that $\approx_U$ is essentially the ground word problem for U[1].

The symbol $s \overset{*}{\Leftrightarrow}_U t$ is used to represent the notion that s and t are joinable modulo U. This relation is defined in figures 2-8 and 2-9.

$\dfrac{}{s \approx_U s}$	**EQ-REFL**
$\dfrac{\{s = t\} \in U}{s \approx_U t}$	**EQ-SUBS**
$\dfrac{t \approx_U s}{s \approx_U t}$	**EQ-SYMM**
$\dfrac{s \approx_U s' \quad s' \approx_U t}{s \approx_U t}$	**EQ-TRAN**
$\dfrac{s \approx_U t}{E_1^+[s] \approx_U E_1^+[t]}$	**EQ-CTXT**

Figure 2-7. Equality Modulo Ground Equations

$\dfrac{s \approx_U t}{\Gamma \rhd s \Rightarrow_U t}$	**JMU-EQU**
$\dfrac{\Gamma \rhd s \to t}{\Gamma \rhd s \Rightarrow_U t}$	**JMU-EVAL**
$\dfrac{\Gamma, Bound(E_1^+) \rhd s \Rightarrow_U t}{\Gamma \rhd E_1^+[s] \Rightarrow_U E_1^+[t]}$	**JMU-CTXT**

Figure 2-8. Inductive Definition of Single Step Joinability Modulo Ground Equations

$$\frac{}{\Gamma \rhd t \overset{0}{\Leftrightarrow}_U t}$$

$$\frac{\Gamma \rhd s \overset{n}{\Leftrightarrow}_U s' \quad \Gamma \rhd s' \Leftrightarrow_U t}{\Gamma \rhd s \overset{n+1}{\Leftrightarrow}_U t}$$

Figure 2-9. Inductive Definition of n-step Joinability Modulo Ground Equations

The symbol $\Leftrightarrow$ represents the symmetric closure of $\Rightarrow$. Using the definitions above, $\Gamma \rhd s \overset{*}{\Leftrightarrow}_U t$ iff $\exists k, \Gamma \rhd s \overset{k}{\Leftrightarrow}_U t$. This convention is used for all transitive relations in this

paper. The joinability modulo ground equations problem can now be defined as follows. Given s, t, and U, is it the case that $\varnothing \triangleright s \overset{*}{\Leftrightarrow}_U t$? Note that $\varnothing$ refers to the empty context.

In terms of OpTT, $\varnothing \triangleright s \overset{*}{\Leftrightarrow}_U t$ means that there exists a proof P of {s=t} where P only uses the symm, trans, cong, and join proof rules. Furthermore, the cong proofs in P only reference the equations in U. So an algorithm that can solve this problem would be able to determine the existence of such a proof in OpTT.

2.2 Inherent Difficulties

Unfortunately, there are some issues which will make it difficult to develop an algorithm which solves the joinability modulo ground equations problem. These difficulties are described in this section.

The first issue is that the problem, as described in the previous section, is undecidable. A special case of this problem is the one in which $U = \varnothing$ and one of the terms does not terminate with respect to the operational semantics. In this case, the problem is equivalent to the problem of determining whether the other term terminates. Since termination is undecidable, joinability modulo ground equations must also be undecidable.

However, if it is possible to design an algorithm that terminates in practical cases and is guaranteed to be correct when it terminates, then that algorithm could still be immensely useful. In the spirit of this practicality, the theorems in this paper will often assume the termination of various algorithms. Likewise, a proof of termination of the parts of the algorithm that are unrelated to evaluation under the operational semantics is beyond the scope of this paper.

2.3 Desired Algorithm Qualities

With the issues described in the previous section in mind, this section attempts to define qualities that the algorithm must possess in order to be practically useful.

First and foremost, the algorithm should terminate in most practical cases, and it should be correct when it terminates. The algorithm presented in this paper is sound and complete with respect to joinability modulo ground equations when it terminates and certain "consistency" requirements for U are met. The algorithm is also conjectured to terminate when all of the terms encountered by the algorithm terminate with respect to evaluation under the operational semantics, but a proof of this conjecture is beyond the scope of this paper.

Next, the algorithm should be efficient. Efficiency is not explicitly defined, and an analysis of the computational complexity of the algorithm is not presented in this paper. However, the algorithm is designed in such a way that searching is minimized or eliminated. The definition of the algorithm allows for a large amount of flexibility in implementation, and efficient implementation suggestions will be provided where appropriate.

Additionally, the algorithm should support the creation of an OpTT proof that equates the supplied terms using the join, symm, trans, and cong proof rules. This means that any hypjoin proof can be replaced with the generated proof of equality, and that proof can then be checked. As a result of this requirement, there is no need to trust the implementation of hypjoin, and hypjoin has no affect on the theoretical properties of OpTT. This goal is accomplished by breaking up the hypjoin algorithm into small steps, each of which results in an equality that can be obtained from symm, join, or cong. These steps are then combined into a single proof using the trans rule.

Finally, in the event that hypjoin fails, an implementation of the algorithm should be able to present some information to the programmer in order to help him determine why hypjoin failed and what must be changed in order for hypjoin to succeed. An algorithm that is based

on searching would probably not satisfy this requirement since the only information that the algorithm could provide is that the search space was exhausted and no suitable match was found. The algorithm presented in this paper is based on normalizing all forms and then testing the normal forms for equality. These normalized terms can then be presented to the programmer, who will use the output to determine why the requested hypjoin operation was not successful.

The qualities in this section describe a sort of loose specification for an algorithm. This specification is referred to as hypjoin. The actual algorithm that is defined in this paper which satisfies this specification will be referred to as the "hypjoin algorithm."

2.4 Additional Notation

This section describes some additional notation that is used in this paper and not explained elsewhere.

Notation such as $\overline{a}$ is used to denote a (possibly empty) list of items $a_1, a_2, \ldots, a_n$ where n is the number of items in $\overline{a}$. A relation on list of items will be used to denote relations on the individual items: $\overline{a}R\overline{b}$ means $\forall n\ a_n R b_n$.

The notation $\Gamma \triangleright a \not R$ indicates that a is normal with respect to relation R in context Γ. The notation $\Gamma \triangleright T(a, R)$ indicates that a terminates with respect to relation R in context Γ. For these notational conventions, Γ is omitted if the relation is not constrained by the context.

The naming conventions of [8] are followed, where possible. For example, s and t usually refer to terms, I refers to an inactive term, and x refers to a variable.

2.5 Example Input

This section contains some example input and the expected results of hypjoin. The purpose of these examples is to explain the expectations of the algorithm and the motivation behind the design of the algorithm.

The example from chapter 1 is a very simple example input set that will be discussed first. The proof using hypjoin is duplicated in figure 2-10 for convenience.

The most complicated hypjoin occurs in the S n' case of the induction. In order to determine that (ge n n) is equivalent to T in this case, hypjoin could begin by realizing that n = (S n') as given by x1, and then determining that (ge n n) = (ge (S n') (S n')). Then hypjoin would determine that (ge (S n') (S n')) = (ge n' n') due to the operational semantics and the definition of ge. Then by transitivity, (ge n n) = (ge n' n'). In these steps, hypjoin simply found substitutions in U that allow the terms to take a step of evaluation in the operational semantics. This "evaluation modulo U" is the basis of the hypjoin algorithm. The term (ge n' n') is considered to be normal because there are no substitutions that would allow it to take a step of evaluation. The same can be said for the term T. From here, hypjoin can simply test the terms (ge n' n') and T for equivalence given substitutions from U. In this case, one of the user-provided equations is {(ge n' n') = T} so the normalized terms are equivalent and hypjoin succeeds.

```
Define geRefl : Forall(n:nat). { (ge n n) = T } :=
        induction(n:nat) by x1 x2 IH return { (ge n n) = T} with
                Z => hypjoin (ge n n) T by x1 end
              | S n' => hypjoin (ge n n) T by x1 [IH n'] end
        end.
```

Figure 2-10. Simple hypjoin Input

Actually, (ge n' n') is not normal w.r.t. the operational semantics – that term will take a step and be replaced with the definition of that function. So the user-provided equations must

also be normalized in order to successfully make the final comparison in the previous example. In fact, each user-provided equation must be evaluated modulo the other equations. The following example describes a case in which this additional evaluation is necessary. This example uses a lemma called geZ which is described in figure 2-11.

The example is a proof of transitivity of ge called geTrans. Two versions of the proof are given: the first version, in figure 2-12, does not use hypjoin. The version in figure 2-13 uses hypjoin in order to simplify the proof.

```
Define geZ : Forall(n : nat)(a1 : {(ge Z n) = T}). {n = Z} :=
        induction(n:nat) by x1 x2 IH return Forall(a1:{(ge Z n) = T}). {n = Z} with
            Z => foralli(a1 : {(ge Z n) = T}).x1
          | S n' => foralli(a1 : {(ge Z n) = T}).
                      contra trans trans symm a1
                                  trans cong (ge Z *) x1
                                        join (ge Z (S n')) F
                                  clash F T
                    {n = Z}
        end.
```

Figure 2-11. Definition of geZ

The definition of geTrans that uses hypjoin illustrates an additional part of this problem that must be addressed. The last use of hypjoin in figure 2-13 attempts to equate (ge b' c') with T, both of which are already normal, and the user-provided equations do not allow any substitutions that can equate these terms. In the standard version of this proof, the term (ge b c) is used to bridge the gap from (ge b' c') to T. In order to do so, it must be determined that (ge b c) = (ge (S b') (S c')), and then (ge (S b') (S c')) can evaluate under the operational semantics to (ge b' c'). The hypjoin algorithm will solve this problem by attempting to normalize each term in the user-provided equations given substitutions allowed by all other equations. By doing so, hypjoin will conclude that (ge b c) can evaluate to (ge b' c'). At this

point, the equation {(ge b c) = T} in U can be replaced with {(ge b' c'} = T} and hypjoin will succeed.

```
Define geTrans : Forall(a b c : nat)
                        (a1 : { (ge a b) = T})
                        (a2 : { (ge b c) = T}). {(ge a c) = T} :=
  induction(a:nat) by x1 x2 IH return Forall(b c : nat)
                                          (a1 : { (ge a b) = T})
                                          (a2 : { (ge b c) = T}).
                                          {(ge a c) = T}  with
    Z => foralli(b c : nat)
                 (a1 : { (ge a b) = T})
                 (a2 : { (ge b c) = T}).
       trans cong (ge a *) [geZ c trans cong (ge * c) symm
                                    geZ b trans cong (ge * b) symm x1 a1]
                           a2]
            join (ge a Z) T
    | S a' => induction(b:nat) by ix1 ix2 IIH return Forall(c : nat)
                                                    (a1 : { (ge a b) = T})
                                                    (a2 : { (ge b c) = T}).
                                                    {(ge a c) = T}  with
       Z => foralli(c : nat)(a1 : { (ge a b) = T})
                                 (a2 : { (ge b c) = T}).
         trans cong (ge a *) [geZ c trans cong (ge * c) symm ix1 a2]
              join (ge a Z) T
      | S b' => induction(c:nat) by iix1 iix2 IIIH return Forall(a1 : { (ge a b) = T})
                                                           (a2 : { (ge b c) = T}).
                                                           {(ge a c) = T}  with
         Z => foralli(a1 : { (ge a b) = T})
                      (a2 : { (ge b c) = T}).
              trans cong (ge a *) iix1
```

```
                    join (ge a Z) T
  | S c' => foralli(a1 : { (ge a b) = T})
                          (a2 : { (ge b c) = T}).
        trans trans trans cong (ge * c) x1
                          cong (ge (S a') *) iix1
                    join (ge (S a') (S c')) (ge a' c')
        [IH a' b' c' trans symm trans trans cong (ge * b) x1
                                          cong (ge (S a') *) ix1
                                    join (ge (S a') (S b')) (ge a' b')
                          a1
                    trans symm trans trans cong (ge * c) ix1
                                          cong (ge (S b') *) iix1
                                    join (ge (S b') (S c')) (ge b' c')
                          a2]
    end
  end
end.
```

Figure 2-12. Standard Definition of geTrans

```
Define geTrans : Forall(a b c : nat)
                          (a1 : { (ge a b) = T})
                          (a2 : { (ge b c) = T}).
                         {(ge a c) = T} :=
  induction(a:nat) by x1 x2 IH return Forall(b c : nat)
                                          (a1 : { (ge a b) = T})
                                          (a2 : { (ge b c) = T}).
                                          {(ge a c) = T}  with
    Z => foralli(b c : nat)(a1 : { (ge a b) = T})
                          (a2 : { (ge b c) = T}).
      hypjoin (ge a c) T by
        [geZ c hypjoin (ge Z c) T by a2
```

```
            [geZ b hypjoin (ge Z b) T by x1 a1 end]
          end]
        end
    | S a' => induction(b:nat) by ix1 ix2 IIH return Forall(c : nat)
                                                    (a1 : { (ge a b) = T})
                                                    (a2 : { (ge b c) = T}).
                                                    {(ge a c) = T}  with
        Z => foralli(c : nat)(a1 : { (ge a b) = T})
                             (a2 : { (ge b c) = T}).
          hypjoin (ge a c) T by
            [geZ c hypjoin (ge Z c) T by a2 ix1 end]
          end
      | S b' => induction(c:nat) by iix1 iix2 IIIH return Forall(a1 : { (ge a b) = T})
                                                    (a2 : { (ge b c) = T}).
                                                    {(ge a c) = T}  with
          Z => foralli(a1 : { (ge a b) = T})
                       (a2 : { (ge b c) = T}).
            hypjoin (ge a c) T by iix1 end
        | S c' => foralli(a1 : { (ge a b) = T})
                         (a2 : { (ge b c) = T}).
          hypjoin (ge a c) T by x1 iix1
            [IH a' b' c'
             hypjoin (ge a' b') T by x1 ix1 a1 end
             hypjoin (ge b' c') T by ix1 iix1 a2 end ]
          end
    end
  end
end.
```

Figure 2-13. Hypjoin Definition of geTrans

The next example shows some of the practical benefit of hypjoin. This example is a proof that addition is associative. The definition of plus is not provided. The standard proof in figure 2-14 is fairly complicated, but each case of the induction uses only trans, symm, cong, and join proof rules. This proof becomes trivial when hypjoin is employed as shown in figure 2-15.

```
Define plus_assoc : Forall(x y z:nat). { (plus (plus x y) z) = (plus x (plus y z)) } :=
        induction(x:nat) by x1 x2 IH return
                    Forall(y z : nat).
                      { (plus (plus x y) z) = (plus x (plus y z)) }
        with
          Z => foralli(y z : nat).
              trans cong (plus (plus * y) z) x1
              trans join (plus (plus Z y) z) (plus y z)
              trans symm join (plus Z (plus y z)) (plus y z)
                   cong (plus * (plus y z)) symm x1
        | S x' => foralli(y z : nat).
                trans cong (plus (plus * y) z) x1
                trans join (plus (plus (S x') y) z) (S (plus (plus x' y) z))
                trans cong (S *) [IH x' y z]
                trans symm join (plus (S x') (plus y z)) (S (plus x' (plus y z)))
                     cong (plus * (plus y z)) symm x1
    end.
```

Figure 2-14. Standard Definition of plus_assoc

The final example illustrates an unusual case that hypjoin is expected to handle correctly. Consider the case in which the programmer would like to hypjoin two terms that will not take a step of evaluation in the operational semantics. For example, the two terms could be fun terms with bodies that can be equated by hypjoin. In this example, ident is the identity function for natural numbers. A proof fragment illustrating this problem is provided in figure 2-16.

```
Define plus_assoc : Forall(x y z:nat). { (plus (plus x y) z) = (plus x (plus y z)) } :=
 induction(x:nat) by x1 x2 IH return
              Forall(y z : nat).
                { (plus (plus x y) z) = (plus x (plus y z)) }
 with
   Z => foralli(y z : nat).
       hypjoin (plus (plus x y) z) (plus x (plus y z)) by x1 end
 | S x' => foralli(y z : nat).
          hypjoin (plus (plus x y) z) (plus x (plus y z)) by x1 [IH x' y z] end
end.
```

Figure 2-15. Hypjoin Definition of plus_assoc

```
hypjoin fun r(a: nat):nat.(a n) fun r(b: nat):nat.(b (ident n)) by end
```

Figure 2-16 Hypjoin of Fun Terms

In order to join the terms in this example, hypjoin must evaluate within the bodies of these fun terms. If the bodies can be equated by hypjoin and the argument types and return type are equal, then the fun terms will be equated by hypjoin. The challenge presented by this example is that hypjoin must preserve the bound variables when evaluating the bodies of these terms. Since (ident n) hypjoins with n and these terms contain no bound variables, hypjoin should succeed.

3 Algorithm Description

This chapter contains a description of an algorithm that solves the problem described in chapter 2. The chapter begins with an overview of the elements of the algorithm and continues with some characteristics that are designed into the algorithm in order to make correctness easier to prove. The next section contains a formal definition of the algorithm, and implementation notes are provided in the final section.

3.1 Algorithm Overview

The algorithm is built around an operation referred to evaluation modulo U, which is represented as $\rightarrow_U$ where U is a set of ground equations. This operation will, in essence, try to find a substitution in U that allows the term to take a step of evaluation with respect to the operational semantics. In the first phase of the algorithm, all of the terms in the equations in U are put into normal form with respect to $\rightarrow_U$. Any time $\rightarrow_U$ is used, it is important that the equations in U are consistent - that is, substitutions in U cannot allow some sort of contradiction. So this normalization of U is structured in such a way that all $\rightarrow_U$ operations make use of normal, consistent U. In the next phase, the two supplied terms are put into normal form with respect to $\rightarrow_U$ using the normal, consistent U that was calculated in the first phase. Finally, the algorithm results in a value of true if the normal forms of the supplied terms are related in $\approx_U$.

3.2 Algorithm Design Considerations

There are a few properties that are very valuable when attempting to prove that the algorithm is correct. The properties described in this section were designed in to the algorithm in order to make the proof of correctness simpler.

First, $\approx_U$ should be preserved by $\rightarrow_U$. That is, if $s \approx_U t$, $s \rightarrow_U s'$, and $t \rightarrow_U t'$, then $s' \approx_U t'$. Another way to view this property is to say that $\rightarrow_U$ is deterministic if the notion

of equality used is $\approx_U$. In order to achieve this determinism, it is necessary to evaluate the subterms of s and t in a specific order. The value of this determinism is that the proof that terms are joined by $\rightarrow_U$ can be a simple inductive proof that appeals to this preservation of $\approx_U$ in each step of evaluation. Without this property, it would be necessary to show that $\rightarrow_U$ is confluent[1], which would be more difficult.

It is also necessary to ensure that the ground equations are consistent in order to prevent a scenario in which the scrutinee of a match term is equivalent to the patterns of multiple cases in the term. For example, if the equations contain {Z = (S n)}, then a match on type Nat could take a step in $\rightarrow_U$ to the bodies of both cases. Because these bodies are not necessarily related by $\approx_U$, the determinism property described previously would be violated in this scenario.

Next, it is necessary for two terms to be related by $\downarrow_U$ if the terms differ only in their strict subterms and those subterms are related by $\downarrow_U$. In othe words, $\bar{a} \downarrow_U \bar{b}$ implies $E_1^*[\bar{a}] \downarrow_U E_1^*[\bar{b}]$. This property is necessary for the completeness of the algorithm. In order to make this property easier to prove, $t \rightarrow_U$ will normalize all of the strict subterms of t with respect to $\rightarrow_U$ before looking for a substitution in U that allows t to take a step in the operational semantics.

3.3 Algorithm Definition

This section provides a formal definition of the hypjoin algorithm.

3.3.1 Evaluation Modulo U

The evaluation modulo U operation ($\rightarrow_U$) defined in figure 3-1 is used in several places in the algorithm. The equivalence relation used in $\rightarrow_U$ is more restrictive than $\approx_U$ as it only allows substitutions in evaluation contexts. This relation is denoted $=_U$ and is defined in fiture 3-2. It is important to note that $=_U \subset \approx_U$.

$\dfrac{}{s =_U s}$	**REQ-REFL**
$\dfrac{\{s = t\} \in U}{s =_U t}$	**REQ-SUBS**
$\dfrac{t =_U s}{s =_U t}$	**REQ-SYMM**
$\dfrac{s =_U s' \quad s' =_U t}{s =_U t}$	**REQ-TRAN**
$\dfrac{s =_U t}{E_1[s] =_U E_1[t]}$	**REQ-CTXT**

Figure 3-1. Restrictive Equality Modulo Ground Equations

$\dfrac{\forall m < n \quad \Gamma, Bound(E_1^*, m) \triangleright a_m \not\rightarrow_U \quad \Gamma, Bound(E_1^*, n) \triangleright a_n \rightarrow_U a_n'}{\Gamma \triangleright E_1^*[\overline{a}] \rightarrow_U E_1^*[a_{1\ldots n-1}, a_n', a_{n+1\ldots Holes(E_1^*)}]}$	**EMU-IND**
$\dfrac{\forall n\, \Gamma, Bound(E_1^*, n) \triangleright a_n \not\rightarrow_U \quad E_1^*[\overline{a}] =_U t' \quad \Gamma \triangleright t' \rightarrow t}{\Gamma \triangleright E_1^*[\overline{a}] \rightarrow_U t}$	**EMU-BASE**

Figure 3-2. Evaluation Modulo U

The definition of $\rightarrow_U$ references a function called Holes which takes a one-level-deep extended context and returns the number of holes in that context.

Note that $\rightarrow_U$ is considered to be a family of unary relations (that is, sets) indexed by the starting term. So $t \rightarrow_U$ is a set of terms that can be reached in one step of $\rightarrow_U$ starting at t. The definitions include a test that some term does not take a step in $\rightarrow_U$ denoted $a \not\rightarrow_U$. In terms of unary relations, $a \not\rightarrow_U$ is defined as $a \rightarrow_U = \varnothing$.

The relation $\rightarrow_U$ is not considered to be a binary relation because that would result in impredicativity[6]. When considering whether $s \rightarrow_U t$, it is necessary to test whether the

subterms of s are related to any other terms in $\rightarrow_U$. This test would be impredicative if $\rightarrow_U$ was a binary relation, because the definition would appeal to the entire relation $\rightarrow_U$ as it is being constructed. Because $\rightarrow_U$ is a family of unary relations, and because the definition of $t \rightarrow_U$ only appeals to the $\rightarrow_U$ of strict subterms of t, $t \rightarrow_U$ is predicative for all t.

3.3.2 Consistency

The correctness of the algorithm depends on the consistency of U. That is, U must not allow some sort of contradiction. The rules for consistency are provided in figure 3-3. All of the consistency rules must hold in order for a given U to be consistent. The abbreviation C(U) is used to describe the property that U is consistent.

$$\frac{fun(x:!).i \approx_U fun(y:!).j}{i \approx_U j[x/y]}$$

$$\frac{C \approx_U D}{C = D} \text{ (C and D are term constructors)}$$

$$\frac{}{C \not\approx_U F} \text{ (C is a term constructor, F is a fun term)}$$

$$\frac{(C\ a) \approx_U (D\ b)}{C = D} \text{ (C and D are term constructors, a and b are inactive terms)}$$

$$\frac{}{(C\ a) \not\approx_U D} \text{ (C and D are term constructors, a is an inactive term)}$$

Figure 3-3. Consistency Rules

The first consistency rule related to fun term equivalence is very restrictive and essentially removes any benefit to including fun terms in U. In practice, this restriction is not very severe since the programmer can simply prove the equality of the bodies of the fun terms and place that equation in U.

3.3.3 Normalization of U

The first phase of the algorithm is the attempted normalization of the terms in the equations in U with respect to $\rightarrow_U$. So it is necessary to define $\rightarrow_U$ for sets of equations. This relation is defined inductively on the cardinality of U as shown in figure 3-4.

$$\frac{}{0 : \varnothing \rightarrow \varnothing}$$

$$\frac{m : U \overset{!}{\rightarrow} U' \quad C(U') \quad \varnothing \rhd u1 \overset{!}{\rightarrow}_U u1'}{m : U \cup^* \{u1 = u2\} \rightarrow U' \cup^* \{u1' = u2\}}$$

$$\frac{m : U \overset{!}{\rightarrow} U' \quad \neg C(U')}{m+1 : U \cup^* \{u1 = u2\} \rightarrow U' \cup^* \{u1 = u2\}}$$

Figure 3-4. Evaluation of U

In the definition of evaluation of U, a disjoint union ($\cup^*$) is used to ensure that the size of U is predictable. In practice, duplicate equations can be removed at any time as they will have no impact on the outcome. Because m is always the size of U, m will often be omitted from the notation. Note that the $\rightarrow$ symbol is overloaded. When applied to terms it refers to evaluation under the operational semantics. When it is applied to sets of ground equations, it refers to the "U evaluation" defined in this section.

3.3.4 The Hypjoin Algorithm

The algorithm which implements hypjoin is represented by the symbol $\downarrow_U$. The definition of $\downarrow_U$ is provided in figure 3-5.

$$\frac{U \overset{!}{\rightarrow} U' \quad C(U') \quad \Gamma \rhd s \overset{!}{\rightarrow}_{U'} s' \quad \Gamma \rhd t \overset{!}{\rightarrow}_{U'} t' \quad s' \approx_{U'} t'}{\Gamma \rhd s \downarrow_U t}$$

Figure 3-5. The Hypjoin Algorithm

3.4 Algorithm Implementation Notes

The description of the hypjoin algorithm is intentionally vague in order to allow some amount of freedom when implementing it. This section provides some information that may be helpful when implementing the algorithm.

3.4.1 Equivalence Relations

The equivalence relation $=_U$ is used in the relation $\rightarrow_U$. This relation might be preferable to the more permissive equivalence relation $\approx_U$ because it reduces the size of the search space when trying to find equivalent terms that will take a step of evaluation. However, it would be equally correct to use $\approx_U$ in $\rightarrow_U$. This is because for all s and t $\frac{s \approx_U t \quad s \neq_U t \quad t \rightarrow}{s = t}$. In order to change a stuck term into a term that will take a step of evaluation, it is necessary to either replace the entire term or change something in an evaluation context.

3.4.2 Deciding Equivalence

It was noted earlier that $\approx_U$ is essentially the ground word problem for U. As a result, one could determine whether two terms are equivalent using congruence closure. It may also be possible to solve this problem using Knuth-Bendix completion[7] using a lexicographic path order[1] on the terms. The primary benefit derived from using completion to solve this problem is that it sets up a framework in which other implementation problems can be solved efficiently. These other problems are described in the next sections.

3.4.3 Deciding Consistency

The task of detecting inconsistency may seem daunting at first because it is necessary to consider all the terms that are equivalent to a given term. However, due to the nature of terms that can be related by $\approx_U$, it is only necessary to check for incompatible terms that are related by transitivity. This can be accomplished by iterating over all combinations of incompatible terms in the equations in U and testing them for equivalence.

It may be possible to determine whether U is consistent much more efficiently using Knuth-Bendix completion. Using completion, it is possible to orient the rewrite rules in such a way that all terms rewrite to inactive terms, if possible. Then it is possible to decide consistency simply by completing U and examining the resulting rewrite rules. If U is inconsistent, then there will be a rule that rewrites some term to an incompatible term.

3.4.4 Evaluation Modulo U

In order to determine if a term can take a step in $\rightarrow_U$, one might attempt to search through the entire substitution space until a redex is found. Unfortunately, this space may be infinite, so this approach will not work in general. In order to work around this problem, a limit could be specified in the implementation that will stop searching after a set amount of comparisons. Such a limit would still allow hypjoin to be sound, but it will no longer be complete. Also, this searching can be very inefficient, and it should be avoided if possible.

It may be possible to use Knuth-Bendix completion to solve this problem as well. The rewrite rules can be oriented in such a way that a term will rewrite to a redex, if possible. Then it is possible to find a redex that is equivalent to a given term, if one exists, by normalizing that term with respect to the rewrite system obtained by completing U. Note that this approach is compatible with the consistency test and the same order can be used in both.

4 Proof of Correctness

This chapter contains a proof that the algorithm described in the previous chapter is partially correct under certain stated assumptions. In essence, this chapter will show that the relations $\overset{*}{\Leftrightarrow}_U$ and $\downarrow_U$ are equal if U is consistent. This proof is divided into two theorems. Section 4.1 contains a theorem that states that $\downarrow_U$ is sound with respect to $\overset{*}{\Leftrightarrow}_U$ -- that is, $\downarrow_U \subseteq \overset{*}{\Leftrightarrow}_U$. Section 4.2 contains a theorem that states that $\downarrow_U$ is complete with respect to $\overset{*}{\Leftrightarrow}_U$ -- that is, $\downarrow_U \supseteq \overset{*}{\Leftrightarrow}_U$ -- if the algorithm deciding $\downarrow_U$ terminates and U is consistent.

4.1 Soundness of Hypjoin Algorithm

This section contains a proof that the algorithm is sound as specified in theorem 4.1. The proof is structured as shown in the dependency graph in figure 4-1.

The most significant portion of this proof is lemma 4.1.2, which shows that if $U \to U'$ and $s \overset{*}{\Leftrightarrow}_{U'} t$, then $s \overset{*}{\Leftrightarrow}_U t$. That is, the algorithm does not gain any additional ability to join terms by putting U in normal form.

Theorem 4.1:

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \rhd t1 \downarrow_U t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2}$$

That is,
$$\begin{matrix} \forall t1, \forall t2, \forall U, \forall \Gamma \\ \forall U', \forall t1', \forall t2' \end{matrix} \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \rhd t1 \overset{!}{\to}_{U'} t1' \quad \Gamma \rhd t2 \overset{!}{\to}_{U'} t2' \quad t1' \approx_{U'} t2'}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2}$$

Choose arbitrary t1, t2, U, Γ, U', t1', and t2'

Assume: $U \overset{!}{\to} U' \quad C(U') \quad \Gamma \rhd t1 \overset{!}{\to}_{U'} t1' \quad \Gamma \rhd t2 \overset{!}{\to}_{U'} t2' \quad t1' \approx_{U'} t2'$

Derive: $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2$

From lemma 4.1.1, $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t1'$ and $\Gamma \rhd t2 \overset{*}{\Leftrightarrow}_{U'} t2'$

Because $t1' \approx_{U'} t2'$ and JMU-EQU, $\Gamma \rhd t1' \overset{*}{\Leftrightarrow}_{U'} t2'$

From transitivity, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$

From lemma 4.1.2, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$

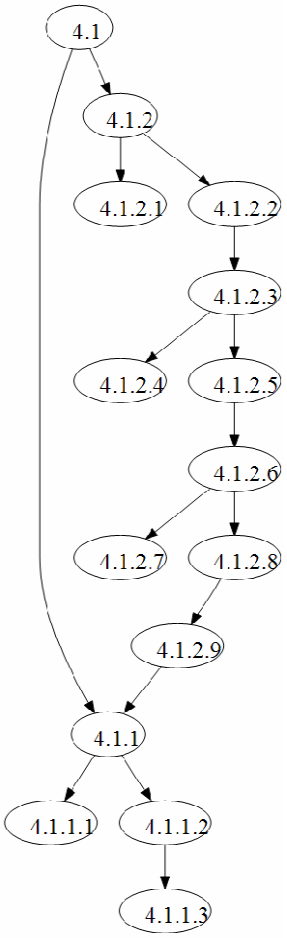

Figure 4-1. Structure of Soundness Proof

Lemma 4.1.1:

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{*}{\rightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}$$

That is: $\forall k, \forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{\Gamma \triangleright t1 \overset{k}{\rightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}$

Proof by induction on k:

- Base case(4.1.1.1): $\forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{\Gamma \triangleright t1 \overset{0}{\rightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}$

- Step case(4.1.1.2): $\forall k \quad \dfrac{\left(\forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{\Gamma \triangleright t1 \overset{k}{\rightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}\right)}{\left(\forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{\Gamma \triangleright t1 \overset{k+1}{\rightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}\right)}$

Lemma 4.1.1.1:

$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{\Gamma \triangleright t1 \overset{0}{\rightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}$

Choose arbitrary t1, t2, U, and Γ.

Assume: $\Gamma \triangleright t1 \overset{0}{\rightarrow}_U t2$

Derive: $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$

Because $\Gamma \triangleright t1 \overset{0}{\rightarrow}_U t2$, t1=t2.

From EQ-REFL and JMU-EQU, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$

Lemma 4.1.1.2:

$\forall k \quad \dfrac{\left(\forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{\Gamma \triangleright t1 \overset{k}{\rightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}\right)}{\left(\forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{\Gamma \triangleright t1 \overset{k+1}{\rightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}\right)}$

Choose arbitrary k.

Assume(A): $\forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{\Gamma \triangleright t1 \overset{k}{\rightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}$

Derive: $\forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{\Gamma \triangleright t1 \overset{k+1}{\rightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}$

Choose arbitrary t1, t2, U, and Γ.

Assume: $\Gamma \rhd t1 \overset{k+1}{\rightarrow}_U t2$

Derive: $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2$

Because, $\Gamma \rhd t1 \overset{k+1}{\rightarrow}_U t2$, there exists s such that $\Gamma \rhd t1 \overset{k}{\rightarrow}_U s$ and $\Gamma \rhd s \rightarrow_U t2$

From A, $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U s$

From lemma 4.1.1.3, $\Gamma \rhd s \overset{*}{\Leftrightarrow}_U t2$

From transitivity, $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2$

Lemma 4.1.1.3:

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \rhd t1 \rightarrow_U t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2}$$

That is: $\forall k, \forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{\Gamma \rhd k : t1 \rightarrow_U t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2}$

Proof by induction on k:

- Base case: $t1 = E_1^*[\overline{a}] \quad \forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \not\rightarrow_U \quad t1 =_U t3 \quad \Gamma \rhd t3 \rightarrow t2$
  - From JMU-EQU, $\Gamma \rhd t1 \Rightarrow_U t3$ and from JMU-EVAL, $\Gamma \rhd t3 \Rightarrow_U t2$
  - From transitivity, $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2$
- Step case:

$t1 = E_1^*[\overline{a}] \quad \forall n < m \quad \Gamma, Bound(E_1^*, n) \rhd a_n \not\rightarrow_U \quad \Gamma, Bound(E_1^*, m) \rhd k : a_m \rightarrow_U a_m'$

  - Where the induction hypothesis (I.H.) is:

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \rhd k : t1 \rightarrow_U t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2}$$

  - From EMU-IND, t2 = $E_1^*[a_{1\ldots n}, a_m', a_{m+1\ldots Holes(E_1^*)}]$
  - From I.H. $\Gamma, Bound(E_1^*, m) \rhd a_m \overset{*}{\Leftrightarrow}_U a_m'$
  - From JMU_CTXT,

$$\Gamma \rhd E_1^*[a_{1\ldots n}, a_m, a_{m+1\ldots Holes(E_1^*)}] \overset{*}{\Leftrightarrow}_U \Gamma \rhd E_1^*[a_{1\ldots n}, a_m', a_{m+1\ldots Holes(E_1^*)}]$$

  - That is, $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2$

Lemma 4.1.2:

$$\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$$

That is: $\forall j, \forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{j}{\Leftrightarrow}_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$

Proof by induction on j.

- Base case: (4.1.2.1) $\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{0}{\Leftrightarrow}_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$
- Step case: (4.1.2.2)

$$\forall j \quad \frac{\left(\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{j}{\Leftrightarrow}_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2} \right)}{\left(\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{j+1}{\Leftrightarrow}_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2} \right)}$$

Lemma 4.1.2.1:

$$\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{0}{\Leftrightarrow}_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$$

So t1=t2

Then $t1 \overset{*}{\Leftrightarrow}_{U} t2$

Lemma 4.1.2.2:

$$\forall j \quad \frac{\left(\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{j}{\Leftrightarrow}_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2} \right)}{\left(\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{j+1}{\Leftrightarrow}_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2} \right)}$$

Choose arbitrary j.

Assume: (A) $\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{j}{\Leftrightarrow}_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$

Derive: $\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\to} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{j+1}{\Leftrightarrow}_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$

Because $\Gamma \triangleright t1 \overset{j+1}{\Leftrightarrow}_{U'} t2$, there exists a t3 such that $\Gamma \triangleright t1 \overset{j}{\Leftrightarrow}_{U'} t3$ and $\Gamma \triangleright t3 \Leftrightarrow_{U'} t2$

From A, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t3$

From lemma 4.1.2.3, $\Gamma \triangleright t3 \overset{*}{\Leftrightarrow}_{U} t2$

From transitivity, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2$

Lemma 4.1.2.3:

$$\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$$

That is: $\forall k, \forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!k}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$

The notation $U \overset{!k}{\rightarrow} U'$ means U normalizes to U' in k steps.

Proof by induction on k:

- Base case: (4.1.2.4) $\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!0}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$
- Step case: (4.1.2.5)

$$\forall k \quad \frac{\left(\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!k}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2} \right)}{\left(\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!k+1}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2} \right)}$$

Lemma 4.1.2.4:

$$\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{U \overset{!0}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$$

So U = U'

Then $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2$ because $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Lemma 4.1.2.5:

$$\forall k \quad \frac{\left(\forall t1,\forall t2,\forall U,\forall U',\forall \Gamma \quad \frac{U \overset{!k}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}\right)}{\left(\forall t1,\forall t2,\forall U,\forall U',\forall \Gamma \quad \frac{U \overset{!k+1}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}\right)}$$

Choose arbitrary k.

Assume(A): $\forall t1,\forall t2,\forall U,\forall U',\forall \Gamma \quad \frac{U \overset{!k}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$

Derive: $\forall t1,\forall t2,\forall U,\forall U',\forall \Gamma \quad \frac{U \overset{!k+1}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$

Because $U \overset{!k+1}{\rightarrow} U'$, there exists a U'' such that $U \rightarrow U''$ and $U'' \overset{!k}{\rightarrow} U'$

From lemma 4.1.2.6, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U''} t2$

From A, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2$

Lemma 4.1.2.6:

$$\forall t1,\forall t2,\forall U,\forall U',\forall \Gamma \quad \frac{U \rightarrow U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$$

That is: $\forall m,\forall t1,\forall t2,\forall U,\forall U',\forall \Gamma \quad \frac{m : U \rightarrow U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$

Proof by induction on m:

- Base case: (4.1.2.7)

  $$\forall t1,\forall t2,\forall U,\forall U',\forall \Gamma \quad \frac{0 : U \rightarrow U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}$$

- Step case: (4.1.2.8)

  $$\forall m \quad \frac{\left(\forall t1,\forall t2,\forall U,\forall U',\forall \Gamma \quad \frac{m : U \rightarrow U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}\right)}{\left(\forall t1,\forall t2,\forall U,\forall U',\forall \Gamma \quad \frac{m+1 : U \rightarrow U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}\right)}$$

Lemma 4.1.2.7:

$$\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{0 : U \to U' \quad C(U') \quad \Gamma \rhd t1 \Leftrightarrow_{U'} t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2}$$

Because $0 : U \to U'$, U=U'=$\varnothing$

So $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2$, because $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Lemma 4.1.2.8:

$$\forall m \quad \frac{\left(\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{m : U \to U' \quad C(U') \quad \Gamma \rhd t1 \Leftrightarrow_{U'} t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2} \right)}{\left(\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{m+1 : U \to U' \quad C(U') \quad \Gamma \rhd t1 \Leftrightarrow_{U'} t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2} \right)}$$

Choose arbitrary m.

Assume(A): $\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{m : U \to U' \quad C(U') \quad \Gamma \rhd t1 \Leftrightarrow_{U'} t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2}$

Derive: $\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{m+1 : U \to U' \quad C(U') \quad \Gamma \rhd t1 \Leftrightarrow_{U'} t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2}$

Only the case where $\Gamma \rhd t1 \Rightarrow_{U'} t2$ will be considered, the proof of the symmetric case is similar and omitted.

Derive: $\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{m+1 : U \to U' \quad C(U') \quad \Gamma \rhd t1 \Rightarrow_{U'} t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2}$

That is: $\forall k, \forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{m+1 : U \to U' \quad C(U') \quad \Gamma \rhd k : t1 \Rightarrow_{U'} t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2}$

Proof by induction on k:

- Base case: $t1 \approx_{U'} t2$
  - From lemma 4.1.2.9, $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2$
- Base case: $\Gamma \rhd t1 \to t2$
  - From JMU-EVAL $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2$
- Step case: $t1 = E_1^+[a1] \quad t2 = E_1^+[a2] \quad \Gamma, Bound(E_1^+) \rhd k : a1 \Rightarrow_{U'} a2$
  - I.H. is $\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{m+1 : U \to U' \quad C(U') \quad \Gamma \rhd k : t1 \Rightarrow_{U'} t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U} t2}$

- From I.H. $\Gamma, Bound(E_1^+) \triangleright a1 \overset{*}{\Leftrightarrow}_U a2$
- From JMU-CTXT, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$

Lemma 4.1.2.9:

$$\frac{\left(\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \dfrac{m : U \to U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}\right)}{\left(\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \dfrac{m+1 : U \to U' \quad C(U') \quad t1 \approx_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}\right)}$$

Assume (A): $\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \dfrac{m : U \to U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}$

Derive: $\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \dfrac{m+1 : U \to U' \quad C(U') \quad t1 \approx_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}$

That is: $\forall k, \forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \dfrac{m+1 : U \to U' \quad C(U') \quad k : t1 \approx_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}$

Because $m+1 : U \to U'$, there exist U_m, U_m', u1, u1', and u2 such that $U = U_m \cup \{u1 = u2\}$, $u1 \to_{U_m'} u1'$, $U' = U_m' \cup \{u1' = u2\}$

Proof by induction on k:

- Base case: t1=t2
  - From JMU-EQU, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$
- Base case: $\{t1 = t2\} \in U'$
  - Case $\{t1 = t2\} \in U_m'$
    - From A, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U_m} t2$
    - So $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$
  - Case t1 = u1' and t2 = u2
    - From lemma 4.1.1, $\Gamma \triangleright u1 \overset{*}{\Leftrightarrow}_{U_m'} u1'$
    - From A, $\Gamma \triangleright u1 \overset{*}{\Leftrightarrow}_{U_m} u1'$
    - So $\Gamma \triangleright u1 \overset{*}{\Leftrightarrow}_U u1'$
    - Because $\{u1 = u2\} \in U$, $\Gamma \triangleright u1 \overset{*}{\Leftrightarrow}_U u2$ (EQ-SUBS, JMU-EQU)

- From transitivity, $\Gamma \triangleright u1' \overset{*}{\Leftrightarrow}_U u2$
- That is, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$

- Step case: $k : t2 \approx_{U'} t1$
  - I.H. (in all step cases) is
    $$\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{m+1 : U \to U' \quad C(U') \quad k : t1 \approx_{U'} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}$$
  - From I.H. $\Gamma \triangleright t2 \overset{*}{\Leftrightarrow}_U t1$
  - From symmetry, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$
- Step case: $k : t1 \approx_{U'} t3$ and $k : t3 \approx_{U'} t2$
  - From I.H. $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t3$
  - From I.H. $\Gamma \triangleright t3 \overset{*}{\Leftrightarrow}_U t2$
  - From transitivity, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$
- Step case: $t1 = E_1^+[a1] \quad t2 = E_1^+[a2] \quad k : a1 \approx_{U'} a2$
  - From I.H. $\Gamma, Bound(E_1^+) \triangleright a1 \overset{*}{\Leftrightarrow}_U a2$
  - From JMU-CTXT, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$

4.2 Completeness of Hypjoin Algorithm

This section contains a proof that the algorithm is complete, that is $s \overset{*}{\Leftrightarrow}_U t$ implies $s \downarrow_U t$, if the algorithm terminates and U does not allow inconsistency. This property is specified and proven in theorem 4.2. The structure of the proof is shown in figure 4-2.

Lemma 4.2.1 shows that if $U \to U'$ and $s \overset{*}{\Leftrightarrow}_U t$, then $s \overset{*}{\Leftrightarrow}_{U'} t$. That is, joinability modulo U is preserved by putting U in normal form. With this lemma in place, the remainder of the proof can assume that U is normal.

Lemma 4.2.3 shows that $s \approx_U t$ implies $s \downarrow_U t$ for normal, consistent U. This lemma is convenient and is therefore used frequently in the proof. Any time it is necessary to prove

that two terms are related in $\downarrow_U$, the proof can show that those two terms are related in $s \approx_U t$.

The property that $\approx_U$ is preserved by $\rightarrow_U$, which was designed into the hypjoin algorithm, is proven in lemma 4.2.5. This lemma is used in the proof of lemma 4.2.3.

Because $s \rightarrow t$ implies $s \Rightarrow_U t$, it is necessary to show that $s \rightarrow t$ implies $s \downarrow_U t$. This property is proven in lemma 4.2.7.

Lemma 4.2.2.6 shows that $\bar{a} \downarrow_U \bar{b}$ implies $E_1^*[\bar{a}] \downarrow_U E_1^*[\bar{b}]$. That is, two terms are related in $\downarrow_U$ if they are the same term at the top level and all of the subterms are related in $\downarrow_U$. This lemma is necessary because $\bar{a} \Rightarrow_U \bar{b}$ implies $E_1^*[\bar{a}] \Rightarrow_U E_1^*[\bar{b}]$ in the definition of $\Rightarrow_U$.

Theorem 4.2:

$$\forall t1, \forall t2, \forall U, \forall U', \forall \Gamma \quad \frac{\Gamma \rhd T(t1, \rightarrow_{U'}) \quad \Gamma \rhd T(t2, \rightarrow_{U'})}{\left(\frac{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2 \quad U \overset{!}{\rightarrow} U' \quad C(U')}{\Gamma \rhd t1 \downarrow_U t2} \right)}$$

Choose arbitrary t1, t2, U, U', and Γ.
Assume: $\Gamma \rhd T(t1, \rightarrow_{U'}) \quad \Gamma \rhd T(t2, \rightarrow_{U'})$
Assume: $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2 \quad U \overset{!}{\rightarrow} U' \quad C(U')$
Derive: $\Gamma \rhd t1 \downarrow_U t2$

Lemma 4.2.1 states: $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{U \overset{!}{\rightarrow} U' \quad C(U') \quad \Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

From Lemma 4.2.1, $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2$.
Lemma 4.2.2 states:

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd T(t1, \rightarrow_U) \quad \Gamma \rhd T(t2, \rightarrow_U)}{\Gamma \rhd t1 \downarrow_U t2}$$

Because $U \overset{!}{\rightarrow} U'$, and $U \not\rightarrow$
From Lemma 4.2.2, $\Gamma \rhd t1 \downarrow_{U'} t2$
So $t1 \downarrow_U t2$ because $t1 \downarrow_{U'} t2$ and $U \overset{!}{\rightarrow} U'$

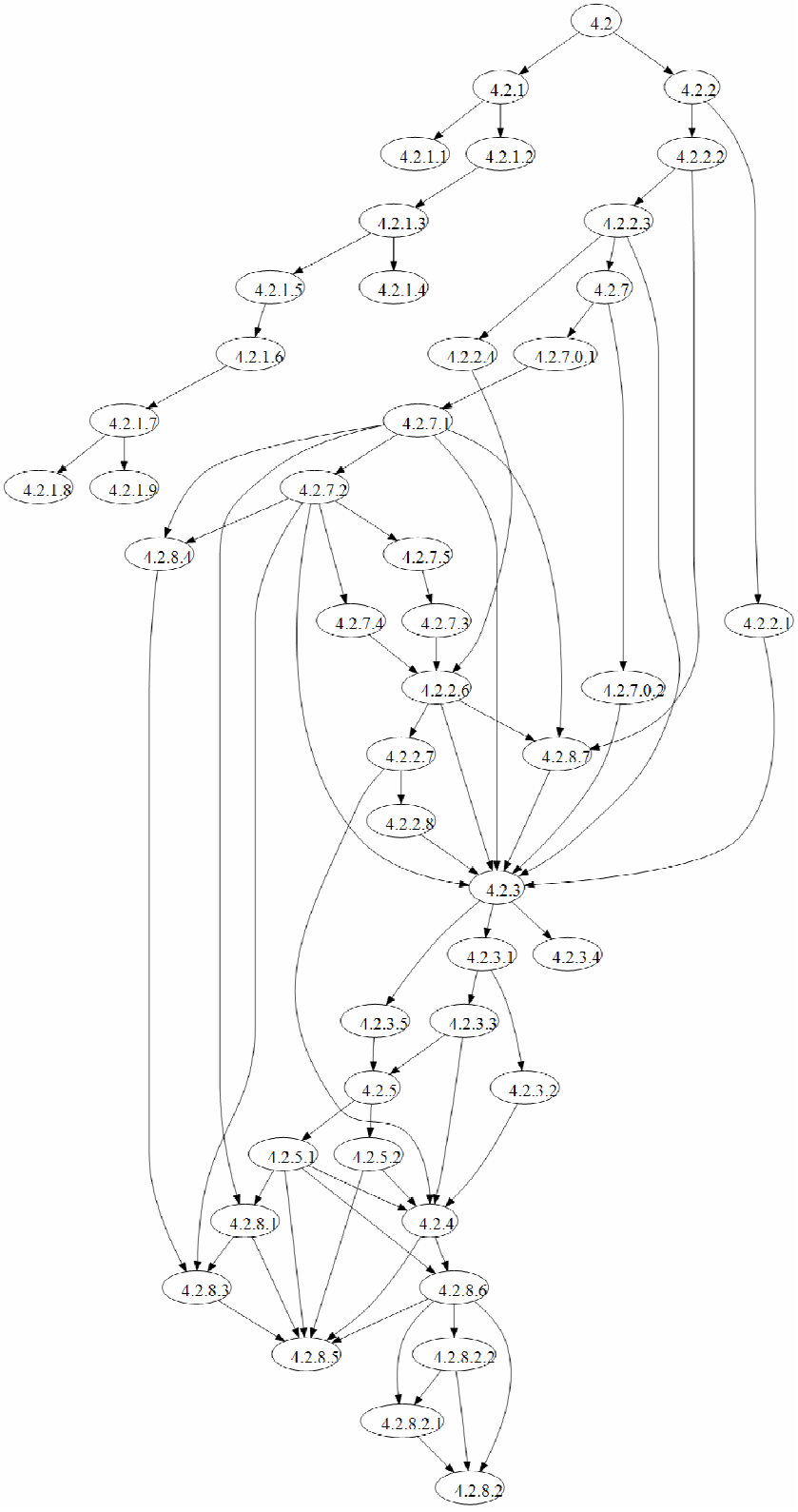

Figure 4-2. Structure of Completeness Proof

Lemma 4.2.1:

$$\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \overset{!}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$$

That is: $\forall n, \forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \overset{!n}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

Proof by induction on n:

- Base case: (4.2.1.1) $\forall n, \forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \overset{!0}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$
- Step case: (4.2.1.2)

$$\forall n \quad \frac{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \overset{!n}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \overset{!n+1}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}$$

Lemma 4.2.1.1:

$$\forall n, \forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \overset{!0}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$$

Choose arbitrary n, U, U', t1, t2, and Γ,

Assume: $U \overset{!0}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2$

Derive: $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Because $U \overset{!0}{\rightarrow} U'$, U=U'

So $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$ because $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2$

Lemma 4.2.1.2:

$$\forall n \quad \frac{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \overset{!n}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \overset{!n+1}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U} t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}$$

Choose arbitrary n.

Assume(A): $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{U \overset{!n}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

Derive: $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{U \overset{!n+1}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

Choose arbitrary U, U', t1, t2, and Γ.

Assume: $U \overset{!n+1}{\rightarrow} U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2$

Derive: $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Because $U \overset{!n+1}{\rightarrow} U'$ and C(U'), there exists U'' such that $U \rightarrow U''$ and $U'' \overset{!n}{\rightarrow} U'$ and C(U'')

From lemma 4.2.1.3, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U''} t2$

From A, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$

<u>Lemma 4.2.1.3:</u>

$\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{U \rightarrow U' \quad C(U') \quad \Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

That is: $\forall k, \forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{U \rightarrow U' \quad C(U') \quad \Gamma \triangleright t1 \overset{k}{\Leftrightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

Proof by induction on k

- Base case: (4.2.1.4) $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{U \rightarrow U' \quad C(U') \quad \Gamma \triangleright t1 \overset{0}{\Leftrightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$
- Step case: (4.2.1.5)

$$\forall k \quad \frac{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{U \rightarrow U' \quad C(U') \quad \Gamma \triangleright t1 \overset{k}{\Leftrightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{U \rightarrow U' \quad C(U') \quad \Gamma \triangleright t1 \overset{k+1}{\Leftrightarrow}_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}$$

Lemma 4.2.1.4:

$$\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \to U' \quad C(U') \quad \Gamma \rhd t1 \overset{0}{\Leftrightarrow}_U t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2}$$

Choose arbitrary U, U', t1, t2, and Γ.

Assume: $U \to U' \quad C(U') \quad \Gamma \rhd t1 \overset{0}{\Leftrightarrow}_U t2$

Derive: $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Because $\Gamma \rhd t1 \overset{0}{\Leftrightarrow}_U t2$, t1=t2

From EQ-REFL and JMU-EQU, $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Lemma 4.2.1.5:

$$\forall k \quad \frac{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \to U' \quad C(U') \quad \Gamma \rhd t1 \overset{k}{\Leftrightarrow}_U t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \to U' \quad C(U') \quad \Gamma \rhd t1 \overset{k+1}{\Leftrightarrow}_U t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}$$

Choose arbitrary k.

Assume(A): $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \to U' \quad C(U') \quad \Gamma \rhd t1 \overset{k}{\Leftrightarrow}_U t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

Derive: $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \to U' \quad C(U') \quad \Gamma \rhd t1 \overset{k+1}{\Leftrightarrow}_U t2}{\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

Choose arbitrary U, U', t1, t2, and Γ.

Assume: $U \to U' \quad C(U') \quad \Gamma \rhd t1 \overset{k+1}{\Leftrightarrow}_U t2$

Derive: $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Because $\Gamma \rhd t1 \overset{k+1}{\Leftrightarrow}_U t2$, there exists t3 such that $\Gamma \rhd t1 \overset{k}{\Leftrightarrow}_U t3$ and $\Gamma \rhd t3 \Leftrightarrow_U t2$

From A, $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t3$

From Lemma 4.2.1.6, $\Gamma \rhd t3 \overset{*}{\Leftrightarrow}_{U'} t2$

From transitivity, $\Gamma \rhd t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Lemma 4.2.1.6:

$$\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \to U' \quad C(U') \quad \Gamma \triangleright t1 \Leftrightarrow_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$$

This proposition will be proven for $\Gamma \triangleright t1 \Rightarrow_U t2$, the proof of $\Gamma \triangleright t1 \Leftarrow_U t2$ similar and omitted.

Proof by induction on the structure of $\Gamma \triangleright t1 \Rightarrow_U t2$

- Case $t1 \approx_U t2$
  - From lemma 4.2.1.7, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$
- Case $\Gamma \triangleright t1 \to t2$
  - From JMU-EVAL $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$
- Case $t1 = E_1^+[a1] \quad t2 = E_1^+[a2] \quad \Gamma, Bound(E_1^+) \triangleright k : a1 \Rightarrow_U a2$
  - I.H. is $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \to U' \quad C(U') \quad \Gamma \triangleright k : t1 \Rightarrow_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$
  - From I.H. $\Gamma, Bound(E_1^+) \triangleright a1 \overset{*}{\Leftrightarrow}_{U'} a2$
  - From JMU-CTXT $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Lemma 4.2.1.7:

$$\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{U \to U' \quad C(U') \quad t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$$

That is, $\forall m, \forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{m : U \to U' \quad C(U') \quad t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

Proof by induction on m:

- Base case(4.2.1.8) $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{0 : U \to U' \quad C(U') \quad t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$
- Step case(4.2.1.9)

$$\forall m \quad \frac{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{m : U \to U' \quad C(U') \quad t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{m+1 : U \to U' \quad C(U') \quad t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}$$

Lemma 4.2.1.8:

$$\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{0 : U \to U' \quad C(U') \quad t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$$

Choose arbitrary U, U', t1, t2, and Γ.
Assume: $0 : U \to U' \quad C(U') \quad t1 \approx_U t2$

Derive: $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Because $0 : U \to U'$, $U = U' = \varnothing$
Because U=U' and $t1 \approx_U t2$, $t1 \approx_{U'} t2$

From JMU-EQU, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Lemma 4.2.1.9:

$$\forall m \quad \frac{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{m : U \to U' \quad C(U') \quad t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}{\left(\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{m+1 : U \to U' \quad C(U') \quad t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2} \right)}$$

Choose arbitrary m.

Assume(A): $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{m : U \to U' \quad C(U') \quad t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

Derive: $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \frac{m+1 : U \to U' \quad C(U') \quad t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$

Choose arbitrary U, U', t1, t2, and Γ.
Assume: $m+1 : U \to U' \quad C(U') \quad t1 \approx_U t2$

Derive: $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Because $m+1 : U \to U'$ and $C(U')$, there exists U_m, U_m', u1, u2, and u1' such that $U = U_m \cup \{u1 = u2\}$, $m : U_m \to U_m'$, C(U_m'), and $\varnothing \triangleright u1 \to_{U_m'} u1'$
So $U' = U_m' \cup \{u1' = u2\}$

Proof by induction on $t1 \approx_U t2$

- Case t1=t2
    - From EQ-REFL and JMU_EQU, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$

- Case $\{t1 = t2\} \in U$
  - Case $\{t1 = t2\} \in U_m$
    - From A, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U_m'} t2$
    - So $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$
  - Case t1=u1 and t2=u2
    - Because $\varnothing \triangleright u1 \rightarrow_{U_m'} u1'$, and theorem 4.1, $\varnothing \triangleright u1 \overset{*}{\Leftrightarrow}_{U_m'} u1'$, so $\varnothing \triangleright u1 \overset{*}{\Leftrightarrow}_{U'} u1'$
    - Because $\{u1' = u2\} \in U'$, $u1' \approx_U u2$ so $\varnothing \triangleright u1' \Leftrightarrow_U u2$ from JMU-EQU
    - From transitivity, $\varnothing \triangleright u1 \overset{*}{\Leftrightarrow}_{U'} u2$, that is, $\varnothing \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$
    - Because $\varnothing \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$ and $(Vars(t1) \cup Vars(t2)) \cap \Gamma = \varnothing$, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$
- Case $k : t2 \approx_U t1$
  - I.H. is $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{m : U \rightarrow U' \quad C(U') \quad k : t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$
  - From I.H. $\Gamma \triangleright t2 \overset{*}{\Leftrightarrow}_{U'} t1$
  - From symmetry, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$
- Case $k : t1 \approx_U t3$ and $k : t3 \approx_U t2$
  - I.H. is $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{m : U \rightarrow U' \quad C(U') \quad k : t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$
  - From I.H. $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t3$ and $\Gamma \triangleright t3 \overset{*}{\Leftrightarrow}_{U'} t2$
  - From transitivity, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$
- Case t1= E_1^+ [a1] and t2= E_1^+ [a2] where $k : a1 \approx_U a2$
  - I.H. is $\forall U, \forall U', \forall t1, \forall t2, \forall \Gamma \quad \dfrac{m : U \rightarrow U' \quad C(U') \quad k : t1 \approx_U t2}{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2}$
  - From I.H. $\Gamma, Bound(E_1^+) \triangleright a1 \overset{*}{\Leftrightarrow}_{U'} a2$
  - From JMU-CTXT, $\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_{U'} t2$

Lemma 4.2.2:

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{*}{\Leftrightarrow}_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright T(t1, \rightarrow_U) \quad \Gamma \triangleright T(t2, \rightarrow_U)}{\Gamma \triangleright t1 \downarrow_U t2}$$

That is:

$$\forall k, \forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{k}{\Leftrightarrow}_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright T(t1, \rightarrow_U) \quad \Gamma \triangleright T(t2, \rightarrow_U)}{\Gamma \triangleright t1 \downarrow_U t2}$$

Proof by induction on k:

- Base case (4.2.2.1)

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{0}{\Leftrightarrow}_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright T(t1, \rightarrow_U) \quad \Gamma \triangleright T(t2, \rightarrow_U)}{\Gamma \triangleright t1 \downarrow_U t2}$$

- Step case (4.2.2.2)

$$\forall k \quad \frac{\left(\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{k}{\Leftrightarrow}_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright T(t1, \rightarrow_U) \quad \Gamma \triangleright T(t2, \rightarrow_U)}{\Gamma \triangleright t1 \downarrow_U t2} \right)}{\left(\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{k+1}{\Leftrightarrow}_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright T(t1, \rightarrow_U) \quad \Gamma \triangleright T(t2, \rightarrow_U)}{\Gamma \triangleright t1 \downarrow_U t2} \right)}$$

Lemma 4.2.2.1:

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{0}{\Leftrightarrow}_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright T(t1, \rightarrow_U) \quad \Gamma \triangleright T(t2, \rightarrow_U)}{\Gamma \triangleright t1 \downarrow_U t2}$$

Choose arbitrary t1, t2, U, and Γ.

Assume: $\Gamma \triangleright t1 \overset{0}{\Leftrightarrow}_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright T(t1, \rightarrow_U) \quad \Gamma \triangleright T(t2, \rightarrow_U)$

Derive: $\Gamma \triangleright t1 \downarrow_U t2$

So t1=t2, so $\Gamma \triangleright t1 \approx_U t2$ from EQ-REFL

From lemma 4.2.3, $\Gamma \triangleright t1 \downarrow_U t2$

Lemma 4.2.2.2

$$\forall k \quad \frac{\left(\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{k}{\Leftrightarrow}_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright T(t1, \rightarrow_U) \quad \Gamma \triangleright T(t2, \rightarrow_U)}{\Gamma \triangleright t1 \downarrow_U t2} \right)}{\left(\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{k+1}{\Leftrightarrow}_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright T(t1, \rightarrow_U) \quad \Gamma \triangleright T(t2, \rightarrow_U)}{\Gamma \triangleright t1 \downarrow_U t2} \right)}$$

Choose arbitrary k.

Assume(A):

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{k}{\Leftrightarrow}_U t2 \quad U \not\to \quad C(U) \quad \Gamma \triangleright T(t1, \to_U) \quad \Gamma \triangleright T(t2, \to_U)}{\Gamma \triangleright t1 \downarrow_U t2}$$

Derive: $\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \overset{k+1}{\Leftrightarrow}_U t2 \quad U \not\to \quad C(U) \quad \Gamma \triangleright T(t1, \to_U) \quad \Gamma \triangleright T(t2, \to_U)}{\Gamma \triangleright t1 \downarrow_U t2}$

Choose arbitrary t1, t2, U, and Γ.

Assume: $\Gamma \triangleright t1 \overset{k+1}{\Leftrightarrow}_U t2 \quad U \not\to \quad C(U) \quad \Gamma \triangleright T(t1, \to_U) \quad \Gamma \triangleright T(t2, \to_U)$

Derive: $\Gamma \triangleright t1 \downarrow_U t2$

Because $\Gamma \triangleright t1 \overset{k+1}{\Leftrightarrow}_U t2$, $\exists t3 \quad \Gamma \triangleright t1 \overset{k}{\Leftrightarrow}_U t3 \quad \Gamma \triangleright t3 \Leftrightarrow_U t2$

It is assumed that $\Gamma \triangleright T(t3, \to_U)$

From A, $\Gamma \triangleright t1 \downarrow_U t3$

From lemma 4.2.2.3, $\Gamma \triangleright t3 \downarrow_U t2$

From lemma 4.2.8.7, $\Gamma \triangleright t1 \downarrow_U t2$

<u>Lemma 4.2.2.3:</u>

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \Leftrightarrow_U t2 \quad U \not\to \quad C(U) \quad \Gamma \triangleright T(t1, \to_U) \quad \Gamma \triangleright T(t2, \to_U)}{\Gamma \triangleright t1 \downarrow_U t2}$$

Case $\Gamma \triangleright t1 \Rightarrow_U t2$ (case $\Gamma \triangleright t1 \Leftarrow_U t2$ is omitted because it is similar)

That is: $\exists k \quad \Gamma \triangleright k : t1 \Rightarrow_U t2$

Proof by induction on k.

- Base cases:
  - (4.2.3)

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad U \not\to \quad C(U) \quad \Gamma \triangleright T(t1, \to_U) \quad \Gamma \triangleright T(t2, \to_U)}{\Gamma \triangleright t1 \downarrow_U t2}$$

  - (4.2.7)

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \to t2 \quad U \not\to \quad C(U) \quad \Gamma \triangleright T(t1, \to_U) \quad \Gamma \triangleright T(t2, \to_U)}{\Gamma \triangleright t1 \downarrow_U t2}$$

- Step case: (4.2.2.4)

$$\forall k \quad \frac{\left(\begin{array}{l}\forall t1, \forall t2, \\ \forall U, \forall \Gamma\end{array} \quad \frac{\Gamma \triangleright k : t1 \Rightarrow_U t2 \quad U \not\to \quad C(U) \quad \Gamma \triangleright T(t1, \to_U) \quad \Gamma \triangleright T(t2, \to_U)}{\Gamma \triangleright t1 \downarrow_U t2}\right)}{\left(\begin{array}{l}\forall t1, \forall t2, \\ \forall U, \forall \Gamma\end{array} \quad \frac{\Gamma \triangleright k+1 : t1 \Rightarrow_U t2 \quad U \not\to \quad C(U) \quad \Gamma \triangleright T(t1, \to_U) \quad \Gamma \triangleright T(t2, \to_U)}{\Gamma \triangleright t1 \downarrow_U t2}\right)}$$

Lemma 4.2.2.4:

$$\forall k \quad \frac{\left(\begin{array}{ll}\forall t1,\forall t2, & \dfrac{\Gamma \rhd k : t1 \Rightarrow_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd T(t1,\rightarrow_U) \quad \Gamma \rhd T(t2,\rightarrow_U)}{\Gamma \rhd t1 \downarrow_U t2} \\ \forall U,\forall \Gamma & \end{array}\right)}{\left(\begin{array}{ll}\forall t1,\forall t2, & \dfrac{\Gamma \rhd k+1 : t1 \Rightarrow_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd T(t1,\rightarrow_U) \quad \Gamma \rhd T(t2,\rightarrow_U)}{\Gamma \rhd t1 \downarrow_U t2} \\ \forall U,\forall \Gamma & \end{array}\right)}$$

Choose arbitrary k.

Assume(A): $\begin{array}{l}\forall t1,\forall t2, \\ \forall U,\forall \Gamma\end{array} \quad \dfrac{\Gamma \rhd k : t1 \Rightarrow_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd T(t1,\rightarrow_U) \quad \Gamma \rhd T(t2,\rightarrow_U)}{\Gamma \rhd t1 \downarrow_U t2}$

Derive: $\begin{array}{l}\forall t1,\forall t2, \\ \forall U,\forall \Gamma\end{array} \quad \dfrac{\Gamma \rhd k+1 : t1 \Rightarrow_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd T(t1,\rightarrow_U) \quad \Gamma \rhd T(t2,\rightarrow_U)}{\Gamma \rhd t1 \downarrow_U t2}$

Choose arbitrary t1, t2, U, and Γ

Assume: $\Gamma \rhd k+1 : t1 \Rightarrow_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd T(t1,\rightarrow_U) \quad \Gamma \rhd T(t2,\rightarrow_U)$

Derive: $\Gamma \rhd t1 \downarrow_U t2$

Because $\Gamma \rhd k+1 : t1 \Rightarrow_U t2$, $t1 = E_1^+[a1] \quad t2 = E_1^+[a2] \quad \Gamma, Bound(E_1^+) \rhd k : a1 \Rightarrow_U a2$

From A, $\Gamma, Bound(E_1^+) \rhd a1 \downarrow_U a2$

From lemma 4.2.2.6, $\Gamma \rhd t1 \downarrow_U t2$

Lemma 4.2.2.6:

$$\forall \bar{a},\forall \bar{b},\forall \Gamma,\forall E_1^*,\forall U \quad \frac{\forall n\, \Gamma, Bound(E_1^*,n) \rhd a_n \downarrow_U b_n \quad U \not\rightarrow \quad C(U)}{\Gamma \rhd E_1^*[\bar{a}] \downarrow_U E_1^*[\bar{b}]}$$

Choose arbitrary $\bar{a}, \bar{b}, \Gamma, E_1^*$, and U

Assume: $\forall n\, \Gamma, Bound(E_1^*,n) \rhd a_n \downarrow_U b_n \quad U \not\rightarrow \quad C(U)$

Derive: $\Gamma \rhd E_1^*[\bar{a}] \downarrow_U E_1^*[\bar{b}]$

Choose arbitrary $\bar{a}'$ and $\bar{b}'$ such that $\forall n\, \Gamma, Bound(E_1^*,n) \rhd a_n \overset{!}{\rightarrow}_U a_n'$ and $\forall n\, \Gamma, Bound(E_1^*,n) \rhd b_n \overset{!}{\rightarrow}_U b_n'$ (termination is assumed).

Because $\forall n\, \Gamma, Bound(E_1^*,n) \rhd a_n \downarrow_U b_n, \quad a_n' \approx_U b_n'$

From EQ-CTXT, $E_1^*[\bar{a}'] \approx_U E_1^*[\bar{b}']$

From lemma 4.2.3, $\Gamma \rhd E_1^*[\bar{a}'] \downarrow_U E_1^*[\bar{b}']$

From lemma 4.2.2.7, $\Gamma \rhd E_1^*[\bar{a}] \downarrow_U E_1^*[\bar{a}']$ and $\Gamma \rhd E_1^*[\bar{b}] \downarrow_U E_1^*[\bar{b}']$

From lemma 4.2.8.7, $\Gamma \rhd E_1^*[\bar{a}] \downarrow_U E_1^*[\bar{b}]$

Lemma 4.2.2.7

$$\forall \bar{a}, \forall \bar{b}, \forall \Gamma, \forall E_1^*, \forall U \quad \frac{\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)}{\Gamma \rhd E_1^*[\bar{a}] \downarrow_U E_1^*[\bar{b}]}$$

That is:

$$\forall j, \forall \bar{a}, \forall \bar{b}, \forall \Gamma, \forall E_1^*, \forall U, \forall t, \forall t' \quad \frac{\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)}{\left(\frac{\Gamma \rhd E_1^*[\bar{a}] \overset{!j}{\rightarrow}_U t \quad \Gamma \rhd E_1^*[\bar{b}] \overset{!}{\rightarrow}_U t'}{t \approx_U t'} \right)}$$

Proof by induction on j

- Base case: $\forall \bar{a}, \forall \bar{b}, \forall \Gamma, \forall E_1^*, \forall U, \forall t, \forall t'$ $\quad \frac{\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)}{\left(\frac{\Gamma \rhd E_1^*[\bar{a}] \overset{!0}{\rightarrow}_U t \quad \Gamma \rhd E_1^*[\bar{b}] \overset{!}{\rightarrow}_U t'}{t \approx_U t'} \right)}$
  - Choose arbitrary $\bar{a}, \bar{b}, \Gamma, E_1^*$, U, t, and t'
  - Assume $\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)$
  - Assume $\Gamma \rhd E_1^*[\bar{a}] \overset{!0}{\rightarrow}_U t \quad \Gamma \rhd E_1^*[\bar{b}] \overset{!}{\rightarrow}_U t'$
  - Derive: $t \approx_U t'$
  - Because $\Gamma \rhd E_1^*[\bar{a}] \overset{!0}{\rightarrow}_U t$, $\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \not\rightarrow_U$ and therefore $\bar{a} = \bar{b}$
  - Then $E_1^*[\bar{a}] = E_1^*[\bar{b}]$
  - From EQ-REFL, $E_1^*[\bar{a}] \approx_U E_1^*[\bar{b}]$
  - From lemma 4.2.4, $\Gamma \rhd E_1^*[\bar{b}] \overset{!0}{\rightarrow}_U t'$
  - Because $\Gamma \rhd E_1^*[\bar{a}] \overset{!0}{\rightarrow}_U t$, $E_1^*[\bar{a}] = t$
  - Because $\Gamma \rhd E_1^*[\bar{b}] \overset{!0}{\rightarrow}_U t'$, $E_1^*[\bar{b}] = t'$
  - Then $t \approx_U t'$ because $E_1^*[\bar{a}] \approx_U E_1^*[\bar{b}]$

- Step case: Prove (4.2.2.8)

$$\forall j \quad \frac{\left(\begin{array}{c}\forall \bar{a}, \forall \bar{b}, \forall \Gamma, \\ \forall E_1^*, \forall U, \forall t, \forall t'\end{array} \quad \frac{\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)}{\left(\frac{\Gamma \rhd E_1^*[\bar{a}] \overset{!j}{\rightarrow}_U t \quad \Gamma \rhd E_1^*[\bar{b}] \overset{!}{\rightarrow}_U t'}{t \approx_U t'}\right)}\right)}{\left(\begin{array}{c}\forall \bar{a}, \forall \bar{b}, \forall \Gamma, \\ \forall E_1^*, \forall U, \forall t, \forall t'\end{array} \quad \frac{\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)}{\left(\frac{\Gamma \rhd E_1^*[\bar{a}] \overset{!j+1}{\rightarrow}_U t \quad \Gamma \rhd E_1^*[\bar{b}] \overset{!}{\rightarrow}_U t'}{t \approx_U t'}\right)}\right)}$$

Lemma 4.2.2.8

$$\forall j \quad \frac{\left(\begin{array}{c}\forall \bar{a}, \forall \bar{b}, \forall \Gamma, \\ \forall E_1^*, \forall U, \forall t, \forall t'\end{array} \quad \frac{\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)}{\left(\frac{\Gamma \rhd E_1^*[\bar{a}] \overset{!j}{\rightarrow}_U t \quad \Gamma \rhd E_1^*[\bar{b}] \overset{!}{\rightarrow}_U t'}{t \approx_U t'}\right)}\right)}{\left(\begin{array}{c}\forall \bar{a}, \forall \bar{b}, \forall \Gamma, \\ \forall E_1^*, \forall U, \forall t, \forall t'\end{array} \quad \frac{\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)}{\left(\frac{\Gamma \rhd E_1^*[\bar{a}] \overset{!j+1}{\rightarrow}_U t \quad \Gamma \rhd E_1^*[\bar{b}] \overset{!}{\rightarrow}_U t'}{t \approx_U t'}\right)}\right)}$$

Choose arbitrary j.
Assume(A):

$$\frac{\begin{array}{c}\forall \bar{a}, \forall \bar{b}, \forall \Gamma, \\ \forall E_1^*, \forall U, \forall t, \forall t'\end{array} \quad \frac{\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)}{\left(\frac{\Gamma \rhd E_1^*[\bar{a}] \overset{!j}{\rightarrow}_U t \quad \Gamma \rhd E_1^*[\bar{b}] \overset{!}{\rightarrow}_U t'}{t \approx_U t'}\right)}}{}$$

Derive:

$$\frac{\begin{array}{c}\forall \bar{a}, \forall \bar{b}, \forall \Gamma, \\ \forall E_1^*, \forall U, \forall t, \forall t'\end{array} \quad \frac{\forall n\, \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)}{\left(\frac{\Gamma \rhd E_1^*[\bar{a}] \overset{!j+1}{\rightarrow}_U t \quad \Gamma \rhd E_1^*[\bar{b}] \overset{!}{\rightarrow}_U t'}{t \approx_U t'}\right)}}{}$$

Choose arbitrary $\bar{a}, \bar{b}, \Gamma, E_1^*$, U, t, and t'

Assume: $\forall n\ \Gamma, Bound(E_1^*, n) \rhd a_n \overset{!}{\rightarrow}_U b_n \quad U \not\rightarrow \quad C(U)$

Assume: $\Gamma \rhd E_1^*[\bar{a}] \overset{!j+1}{\rightarrow}_U t \quad \Gamma \rhd E_1^*[\bar{b}] \overset{!}{\rightarrow}_U t'$

Derive: $t \approx_U t'$

Because $\Gamma \rhd E_1^*[\bar{a}] \overset{!j+1}{\rightarrow}_U t$, there exists w such that $\Gamma \rhd E_1^*[\bar{a}] \rightarrow_U w$ and $\Gamma \rhd w \overset{!j}{\rightarrow}_U t$

Case split on the form of $\Gamma \rhd E_1^*[\bar{a}] \rightarrow_U w$

- Case $\forall n\ \Gamma, Bound(E_1^*, n) \rhd a_n \not\rightarrow_U$ and $E_1^*[\bar{a}] =_U w' \quad \Gamma \rhd w' \rightarrow w$
  - Because $\forall n\ \Gamma, Bound(E_1^*, n) \rhd a_n \not\rightarrow_U$, $\bar{a} = \bar{b}$
  - Then $E_1^*[\bar{a}] \approx_U E_1^*[\bar{b}]$
  - From lemma 4.2.3, $t \approx_U t'$
- Case $\forall n < m \quad \Gamma, Bound(E_1^*, n) \rhd a_n \not\rightarrow_U \quad \Gamma, Bound(E_1^*, m) \rhd a_m \rightarrow_U a_m'$
  - Let $\bar{a}'$ be $a_{1\ldots n}, a_m', a_{m+1\ldots Holes(E_1^*)}$
  - So w = $E_1^*[\bar{a}']$ where $\forall n \quad \Gamma, Bound(E_1^*, n) \rhd a_n' \overset{!}{\rightarrow}_U b_n$
  - From A, $t \approx_U t'$

Lemma 4.2.3:

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd T(t1, \rightarrow_U) \quad \Gamma \rhd T(t2, \rightarrow_U)}{\Gamma \rhd t1 \downarrow_U t2}$$

That is:

$$\forall j, \forall k, \forall t1, \forall t2, \forall t1', \forall t2', \forall U, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd t1 \overset{!j}{\rightarrow}_U t1' \quad \Gamma \rhd t2 \overset{!k}{\rightarrow}_U t2'}{t1' \approx_U t2'}$$

From Lemma 4.2.3.1, j=k

So it is sufficient to prove:

$$\forall k, \forall t1, \forall t2, \forall t1', \forall t2', \forall U, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd t1 \overset{!k}{\rightarrow}_U t1' \quad \Gamma \rhd t2 \overset{!k}{\rightarrow}_U t2'}{t1' \approx_U t2'}$$

Proof by induction on k.

- Base case (4.2.3.4)

$$\forall t1, \forall t2, \forall t1', \forall t2', \forall U, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd t1 \overset{!0}{\rightarrow}_U t1' \quad \Gamma \rhd t2 \overset{!0}{\rightarrow}_U t2'}{t1' \approx_U t2'}$$

- Step case (4.2.3.5)

$$\forall k \quad \frac{\left(\begin{array}{c}\forall t1,\forall t2,\\ \forall t1',\forall t2',\forall U,\forall \Gamma\end{array} \quad \frac{t1\approx_U t2 \quad U\not\downarrow \quad C(U) \quad \Gamma\triangleright t1\overset{!k}{\rightarrow}_U t1' \quad \Gamma\triangleright t2\overset{!k}{\rightarrow}_U t2'}{t1'\approx_U t2'}\right)}{\left(\begin{array}{c}\forall t1,\forall t2,\\ \forall t1',\forall t2',\forall U,\forall \Gamma\end{array} \quad \frac{t1\approx_U t2 \quad U\not\downarrow \quad C(U) \quad \Gamma\triangleright t1\overset{!k+1}{\rightarrow}_U t1' \quad \Gamma\triangleright t2\overset{!k+1}{\rightarrow}_U t2'}{t1'\approx_U t2'}\right)}$$

Lemma 4.2.3.1:

$$\begin{array}{c}\forall j,\forall k,\forall U,\forall \Gamma\\ \forall t1,\forall t2,\forall t1',\forall t2'\end{array} \quad \frac{t1\approx_U t2 \quad U\not\downarrow \quad C(U) \quad \Gamma\triangleright t1\overset{!j}{\rightarrow}_U t1' \quad \Gamma\triangleright t2\overset{!k}{\rightarrow}_U t2'}{j=k}$$

Proof by induction on j:

- Base case (4.2.3.2):

$$\begin{array}{c}\forall k,\forall U,\forall \Gamma\\ \forall t1,\forall t2,\forall t1',\forall t2'\end{array} \quad \frac{t1\approx_U t2 \quad U\not\downarrow \quad C(U) \quad \Gamma\triangleright t1\overset{!0}{\rightarrow}_U t1' \quad \Gamma\triangleright t2\overset{!k}{\rightarrow}_U t2'}{k=0}$$

- Step case (4.2.3.3):

$$\forall j \quad \frac{\left(\begin{array}{c}\forall k,\forall U,\forall \Gamma\\ \forall t1,\forall t2,\forall t1',\forall t2'\end{array} \quad \frac{t1\approx_U t2 \quad U\not\downarrow \quad C(U) \quad \Gamma\triangleright t1\overset{!j}{\rightarrow}_U t1' \quad \Gamma\triangleright t2\overset{!k}{\rightarrow}_U t2'}{j=k}\right)}{\left(\begin{array}{c}\forall k,\forall U,\forall \Gamma\\ \forall t1,\forall t2,\forall t1',\forall t2'\end{array} \quad \frac{t1\approx_U t2 \quad U\not\downarrow \quad C(U) \quad \Gamma\triangleright t1\overset{!j+1}{\rightarrow}_U t1' \quad \Gamma\triangleright t2\overset{!k}{\rightarrow}_U t2'}{j+1=k}\right)}$$

Lemma 4.2.3.2:

$$\begin{array}{c}\forall k,\forall U,\forall \Gamma\\ \forall t1,\forall t2,\forall t1',\forall t2'\end{array} \quad \frac{t1\approx_U t2 \quad U\not\downarrow \quad C(U) \quad \Gamma\triangleright t1\overset{!0}{\rightarrow}_U t1' \quad \Gamma\triangleright t2\overset{!k}{\rightarrow}_U t2'}{k=0}$$

Choose arbitrary k, U, Γ, t1, t2, t1', and t2'

Assume: $t1\approx_U t2 \quad U\not\downarrow \quad C(U) \quad \Gamma\triangleright t1\overset{!0}{\rightarrow}_U t1' \quad \Gamma\triangleright t2\overset{!k}{\rightarrow}_U t2'$

Derive: k=0

Because $\Gamma\triangleright t1\overset{!0}{\rightarrow}_U t1'$, $\Gamma\triangleright t1\not\rightarrow_U$

From contrapositive of lemma 4.2.4, $\neg\left(t1\approx_U t2 \quad U\not\downarrow \quad C(U) \quad \exists t2'' \quad \Gamma\triangleright t2\rightarrow_U t2''\right)$

Because $t1\approx_U t2 \quad U\not\downarrow \quad C(U)$, $\neg\left(\exists t2'' \quad \Gamma\triangleright t2\rightarrow_U t2''\right)$

Therefore $\Gamma\triangleright t2\overset{!0}{\rightarrow}_U t2'$

So k=0

Lemma 4.2.3.3:

$$\forall j \quad \frac{\left(\begin{array}{c}\forall k, \forall U, \forall \Gamma \\ \forall t1, \forall t2, \forall t1', \forall t2'\end{array} \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd t1 \overset{!j}{\rightarrow}_U t1' \quad \Gamma \rhd t2 \overset{!k}{\rightarrow}_U t2'}{j = k}\right)}{\left(\begin{array}{c}\forall k, \forall U, \forall \Gamma \\ \forall t1, \forall t2, \forall t1', \forall t2'\end{array} \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd t1 \overset{!j+1}{\rightarrow}_U t1' \quad \Gamma \rhd t2 \overset{!k}{\rightarrow}_U t2'}{j+1 = k}\right)}$$

Choose arbitrary j.

Assume(A): $\begin{array}{c}\forall k, \forall U, \forall \Gamma \\ \forall t1, \forall t2, \forall t1', \forall t2'\end{array} \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd t1 \overset{!j}{\rightarrow}_U t1' \quad \Gamma \rhd t2 \overset{!k}{\rightarrow}_U t2'}{j = k}$

Derive: $\begin{array}{c}\forall k, \forall U, \forall \Gamma \\ \forall t1, \forall t2, \forall t1', \forall t2'\end{array} \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd t1 \overset{!j+1}{\rightarrow}_U t1' \quad \Gamma \rhd t2 \overset{!k}{\rightarrow}_U t2'}{j+1 = k}$

Choose arbitrary k, U, Γ, t1, t2, t1', and t2'

Assume: $t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd t1 \overset{!j+1}{\rightarrow}_U t1' \quad \Gamma \rhd t2 \overset{!k}{\rightarrow}_U t2'$

Derive: j+1=k

Because $\Gamma \rhd t1 \overset{!j+1}{\rightarrow}_U t1'$, there exists t1'' such that $\Gamma \rhd t1 \rightarrow_U t1''$ and $\Gamma \rhd t1'' \overset{!j}{\rightarrow}_U t1'$

From lemma 4.2.4, $\exists t2'' \quad \Gamma \rhd t2 \rightarrow_U t2''$ and $\Gamma \rhd t2'' \overset{!k-1}{\rightarrow}_U t2'$

From lemma 4.2.5, $t1'' \approx_U t2''$

From A, j=k-1

Then j+1=k

Lemma 4.2.3.4:

$$\begin{array}{c}\forall t1, \forall t2, \\ \forall t1', \forall t2', \forall U, \forall \Gamma\end{array} \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd t1 \overset{!0}{\rightarrow}_U t1' \quad \Gamma \rhd t2 \overset{!0}{\rightarrow}_U t2'}{t1' \approx_U t2'}$$

Choose arbitrary t1, t2, t1', t2', U, and Γ.

Assume: $t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \rhd t1 \overset{!0}{\rightarrow}_U t1' \quad \Gamma \rhd t2 \overset{!0}{\rightarrow}_U t2'$

Derive: $t1' \approx_U t2'$

Because $\Gamma \rhd t1 \overset{!0}{\rightarrow}_U t1'$, t1=t1'

Because $\Gamma \rhd t2 \overset{!0}{\rightarrow}_U t2'$, t2=t2'

Then $t1' \approx_U t2'$ because $t1 \approx_U t2$

Lemma 4.2.3.5:

$$\forall k \quad \frac{\left(\begin{array}{cc} \forall t1, \forall t2, & \dfrac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright t1 \overset{!k}{\rightarrow}_U t1' \quad \Gamma \triangleright t2 \overset{!k}{\rightarrow}_U t2'}{t1' \approx_U t2'} \\ \forall t1', \forall t2', \forall U, \forall \Gamma & \end{array}\right)}{\left(\begin{array}{cc} \forall t1, \forall t2, & \dfrac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright t1 \overset{!k+1}{\rightarrow}_U t1' \quad \Gamma \triangleright t2 \overset{!k+1}{\rightarrow}_U t2'}{t1' \approx_U t2'} \\ \forall t1', \forall t2', \forall U, \forall \Gamma & \end{array}\right)}$$

Choose arbitrary k.

Assume(A): $\dfrac{\forall t1, \forall t2, \quad \forall t1', \forall t2', \forall U, \forall \Gamma}{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright t1 \overset{!k}{\rightarrow}_U t1' \quad \Gamma \triangleright t2 \overset{!k}{\rightarrow}_U t2'}{t1' \approx_U t2'}$

Derive: $\dfrac{\forall t1, \forall t2, \quad \forall t1', \forall t2', \forall U, \forall \Gamma}{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright t1 \overset{!k+1}{\rightarrow}_U t1' \quad \Gamma \triangleright t2 \overset{!k+1}{\rightarrow}_U t2'}{t1' \approx_U t2'}$

Choose arbitrary t1, t2, t1', t2', U, and Γ.

Assume: $t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright t1 \overset{!k+1}{\rightarrow}_U t1' \quad \Gamma \triangleright t2 \overset{!k+1}{\rightarrow}_U t2'$

Derive: $t1' \approx_U t2'$

Because $\Gamma \triangleright t1 \overset{!k+1}{\rightarrow}_U t1'$, there exists t1'' such that $\Gamma \triangleright t1 \rightarrow_U t1''$ and $\Gamma \triangleright t1'' \overset{!k}{\rightarrow}_U t1'$

Because $\Gamma \triangleright t2 \overset{!k+1}{\rightarrow}_U t2'$, there exists t2'' such that $\Gamma \triangleright t2 \rightarrow_U t2''$ and $\Gamma \triangleright t2'' \overset{!k}{\rightarrow}_U t2'$

From lemma 4.2.5, $t1'' \approx_U t2''$

From A, $t1' \approx_U t2'$

Lemma 4.2.4:

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \exists t1' \quad \Gamma \triangleright t1 \rightarrow_U t1'}{\exists t2' \quad \Gamma \triangleright t2 \rightarrow_U t2'}$$

Choose arbitrary t1, t2, U, and Γ

Assume: $t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \exists t1' \quad \Gamma \triangleright t1 \rightarrow_U t1'$

Choose arbitrary t1' such that $\Gamma \triangleright t1 \rightarrow_U t1'$

Derive: $\exists t2' \quad \Gamma \triangleright t2 \rightarrow_U t2'$

Proof by induction on the structure of $\Gamma \triangleright t1 \rightarrow_U t1'$

- Case $t1 = E_1^*[\bar{a}] \quad \forall n\, \Gamma, Bound(E_1^*, n) \triangleright a_n \not\rightarrow_U \quad t1 =_U t1'' \quad \Gamma \triangleright t1'' \rightarrow t1'$
  - From lemma 4.2.8.6, $\exists t2'', t2''' \quad t2 =_U t2'' \quad \Gamma \triangleright t2'' \rightarrow t2'''$
  - So $\exists t2' \quad \Gamma \triangleright t2 \rightarrow_U t2'$

- Case $t1 = E_1^*[a_{1\ldots n-1}, a_n, a_{n+1\ldots}] \quad \Gamma, Bound(E_1^*, n) \triangleright k : a_n \rightarrow_U d$
  - I.H. is $\forall t1, \forall t2, \forall t1', \forall U, \forall \Gamma \quad \dfrac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright k : t1 \rightarrow_U t1'}{\exists t2' \quad \Gamma \triangleright t2 \rightarrow_U t2'}$
  - From lemma 4.2.8.5, $t2 = E_1^*[a_{1\ldots n-1}', a_n', a_{n+1\ldots}']$ for some $\bar{a}'$
  - Because $t1 \approx_U t2$, $\forall n \quad a_n \approx_U a_n'$
  - From I.H., $\Gamma, Bound(E_1^*, n) \triangleright a_n' \rightarrow_U d'$ for some d'
  - From EMU-IND, $\exists t2' \quad \Gamma \triangleright t2 \rightarrow_U t2'$

Lemma 4.2.5:

$$\forall t1, \forall t2, \forall t1', \forall t2', \forall U, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright t1 \rightarrow_U t1' \quad \Gamma \triangleright t2 \rightarrow_U t2'}{t1' \approx_U t2'}$$

That is,

$$\forall j, \forall t1, \forall t2, \forall t1', \forall t2', \forall U, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright j : t1 \rightarrow_U t1' \quad \Gamma \triangleright t2 \rightarrow_U t2'}{t1' \approx_U t2'}$$

Proof by induction on j.

- Base case: (4.2.5.1)

$$\begin{array}{l}\forall t1, \forall t2, \forall t1' \\ \forall t2', \forall U, \forall \Gamma\end{array} \quad \frac{t1 = E_1^*[\bar{a}] \quad \forall n\, \Gamma, Bound(E_1^*, n) \triangleright a_n \not\rightarrow_U \quad t1 =_U t1'' \quad \Gamma \triangleright t1'' \rightarrow t1'}{\left(\dfrac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright t2 \rightarrow_U t2'}{t1' \approx_U t2'}\right)}$$

- Step case: (4.2.5.2)

$$\forall j \quad \frac{\left(\begin{array}{l}\forall t1, \forall t2, \forall t1', \\ \forall t2', \forall U, \forall \Gamma\end{array} \quad \dfrac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright j : t1 \rightarrow_U t1' \quad \Gamma \triangleright t2 \rightarrow_U t2'}{t1' \approx_U t2'}\right)}{\left(\begin{array}{l}\forall t1, \forall t2, \forall t1', \\ \forall t2', \forall U, \forall \Gamma\end{array} \quad \dfrac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright j+1 : t1 \rightarrow_U t1' \quad \Gamma \triangleright t2 \rightarrow_U t2'}{t1' \approx_U t2'}\right)}$$

Lemma 4.2.5.1:

$$\begin{array}{l}\forall t1, \forall t2, \forall t1' \\ \forall t2', \forall U, \forall \Gamma\end{array} \quad \frac{t1 = E_1^*[\bar{a}] \quad \forall n\, \Gamma, Bound(E_1^*, n) \triangleright a_n \not\rightarrow_U \quad t1 =_U t1'' \quad \Gamma \triangleright t1'' \rightarrow t1'}{\left(\dfrac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright t2 \rightarrow_U t2'}{t1' \approx_U t2'}\right)}$$

Choose arbitrary t1, t2, t1', t2', U, and Γ.
Assume:

$t1 = E_1^*[\bar{a}] \quad \forall n\, \Gamma, Bound(E_1^*, n) \triangleright a_n \not\rightarrow_U \quad t1 =_U t1'' \quad \Gamma \triangleright t1'' \rightarrow t1'$

Assume: $t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright t2 \rightarrow_U t2'$

Derive: $t1' \approx_U t2'$

From lemma 4.2.8.5, $t2 = E_1^*[\bar{b}]$ for some $\bar{b}$

Because $t1 \approx_U t2$, $\forall n \quad a_n \approx_U b_n$

From the contrapositive of lemma 4.2.4, $\forall n\ \Gamma, Bound(E_1^*, n) \triangleright b_n \not\rightarrow_U$

From lemma 4.2.8.6, $\exists t2"\quad t2 =_U t2"\quad \Gamma \triangleright t2" \rightarrow t2'"$
Then $\Gamma \triangleright t2 \rightarrow_U t2'$ by EMU-BASE only and t2'''=t2'
By transitivity, $t1" \approx_U t2"$
From lemma 4.2.8.1, $t1' \approx_U t2'$

Lemma 4.2.5.2:

$$\forall j\ \frac{\left(\begin{array}{l}\forall t1, \forall t2, \forall t1', \\ \forall t2', \forall U, \forall \Gamma\end{array}\ \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright j : t1 \rightarrow_U t1' \quad \Gamma \triangleright t2 \rightarrow_U t2'}{t1' \approx_U t2'}\right)}{\left(\begin{array}{l}\forall t1, \forall t2, \forall t1', \\ \forall t2', \forall U, \forall \Gamma\end{array}\ \frac{t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright j+1 : t1 \rightarrow_U t1' \quad \Gamma \triangleright t2 \rightarrow_U t2'}{t1' \approx_U t2'}\right)}$$

Choose arbitrary j

Assume(A): $\dfrac{\begin{array}{l}\forall t1, \forall t2, \forall t1', \\ \forall t2', \forall U, \forall \Gamma\end{array}\quad t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright j : t1 \rightarrow_U t1' \quad \Gamma \triangleright t2 \rightarrow_U t2'}{t1' \approx_U t2'}$

Derive: $\dfrac{\begin{array}{l}\forall t1, \forall t2, \forall t1', \\ \forall t2', \forall U, \forall \Gamma\end{array}\quad t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright j+1 : t1 \rightarrow_U t1' \quad \Gamma \triangleright t2 \rightarrow_U t2'}{t1' \approx_U t2'}$

Choose arbitrary t1, t2, t1', t2', U, and Γ
Assume: $t1 \approx_U t2 \quad U \not\rightarrow \quad C(U) \quad \Gamma \triangleright j : t1 \rightarrow_U t1' \quad \Gamma \triangleright t2 \rightarrow_U t2'$
Derive: $t1' \approx_U t2'$

Because $\Gamma \triangleright j+1 : t1 \rightarrow_U t1'$, $t1 = E_1^*[a_{1\ldots n-1}, a_n, a_{n+1\ldots}]$ where
$\forall m < n\ \Gamma, Bound(E_1^*, m) \triangleright a_m \not\rightarrow_U$ and $\Gamma, Bound(E_1^*, n) \triangleright j : a_n \rightarrow_U d$ for some d.
So $t1' = E_1^*[a_{1\ldots n-1}, d, a_{n+1\ldots}]$

From lemma 4.2.8.5, $t2 = E_1^*[\bar{a}']$ for some $\bar{a}'$
Because $t1 \approx_U t2$, $\forall k \quad a_k \approx_U a_k'$
From contrapositive of lemma 4.2.4, $\forall m < n\ \Gamma, Bound(E_1^*, m) \triangleright a_m' \not\rightarrow_U$
From lemma 4.2.4, $\Gamma, Bound(E_1^*, n) \triangleright a_n' \rightarrow_U d'$ for some d'
From A, $d \approx_U d'$

Because $\forall m < n\ \Gamma, Bound(E_1^*, m) \triangleright a_m' \not\rightarrow_U$ and $\Gamma, Bound(E_1^*, n) \triangleright a_n' \rightarrow_U d'$, and
EMU-IND, $t2' = E_1^*[a_{1\ldots n-1}', d', a_{n+1\ldots}']$
From EQ-CTXT, $t1' \approx_U t2'$

Lemma 4.2.7:

$$\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd t1 \to t2 \quad U \not\twoheadrightarrow \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)}$$

Proof by induction on $t1 \to t2$

- Base case:(4.2.7.0.1)

$$\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd 1 : t1 \to t2 \quad U \not\twoheadrightarrow \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)}$$

- Step case:(4.2.7.0.2)

$$\forall n \quad \frac{\left(\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd n : t1 \to t2 \quad U \not\twoheadrightarrow \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)} \right)}{\left(\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd n+1 : t1 \to t2 \quad U \not\twoheadrightarrow \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)} \right)}$$

Lemma 4.2.7.0.1:

$$\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd 1 : t1 \to t2 \quad U \not\twoheadrightarrow \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)}$$

That is:

$$\forall j, \forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd 1 : t1 \to t2 \quad U \not\twoheadrightarrow \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!j}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)}$$

Proof by induction on j:

- Base case:

$$\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd 1 : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!0}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)}$$

  - That is $\Gamma \rhd t1 \not\to_U$
  - The above contradicts $\Gamma \rhd t1 \to t2$
  - So conclude $t1' \approx_U t2'$

- Step case (4.2.7.1)

$$\forall j \quad \frac{\left(\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd 1 : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!j}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)} \right)}{\left(\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd 1 : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!j+1}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)} \right)}$$

Lemma 4.2.7.0.2:

$$\forall n \quad \frac{\left(\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd n : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)} \right)}{\left(\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd n+1 : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\frac{\Gamma \rhd t1 \overset{!}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)} \right)}$$

Choose arbitrary n.

Assume (A): $\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma$ $\dfrac{\Gamma \rhd n : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\dfrac{\Gamma \rhd t1 \overset{!}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)}$

Derive: $\forall t1, \forall t1', \forall t2, \forall t2', \forall U, \forall \Gamma$ $\dfrac{\Gamma \rhd n+1 : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\dfrac{\Gamma \rhd t1 \overset{!}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'} \right)}$

Choose arbitrary t1, t1', t2, t2', U, and Γ.

Assume: $\Gamma \rhd n+1 : t1 \to t2 \quad U \not\to \quad C(U)$

Assume: $\Gamma \rhd t1 \overset{!}{\to}_U t1' \quad \Gamma \rhd t2 \overset{!}{\to}_U t2'$

Derive: $t1' \approx_U t2'$

Because $\Gamma \rhd n+1 : t1 \to t2$, $t1 = E_1[r1]$, $t2 = E_1[r2]$, $\Gamma \rhd : n : r1 \to r2$

More generally, $t1 = E_1^*[a_{1\ldots n-1}, r1, a_{n+1\ldots}]$ and $t2 = E_1^*[a_{1\ldots n-1}, r2, a_{n+1\ldots}]$

Because r1 (which is in position n) is in the evaluation hole of E_1^*, $Bound(E_1^*, n) = \varnothing$

Consider the term $t3 = E_1^*[a_{1\ldots n-1}', r3, a_{n+1\ldots}']$ where

$\forall m \neq n \quad \Gamma, Bound(E_1^*, m) \rhd a_m \overset{!}{\to}_U a_m'$ and $\Gamma \rhd r1 \overset{!}{\to}_U r3$

Because $\to_U$ normalizes all subterms first, $\Gamma \rhd t1 \overset{*}{\to}_U t3 \overset{!}{\to}_U t1'$

Consider the term $t4 = E_1^*[a_{1\ldots n-1}', r4, a_{n+1\ldots}']$ where

$\forall m \neq n \quad \Gamma, Bound(E_1^*, m) \rhd a_m \overset{!}{\to}_U a_m'$ and $\Gamma \rhd r2 \overset{!}{\to}_U r4$

Because $\to_U$ normalizes all subterms first, $\Gamma \rhd t2 \overset{*}{\to}_U t4 \overset{!}{\to}_U t2'$

From A, $r3 \approx_U r4$

From EQ-CTXT, $t3 \approx_U t4$

From lemma 4.2.3, $t1' \approx_U t2'$

Lemma 4.2.7.1:

$$\forall j \quad \frac{\left(\begin{array}{l}\forall t1, \forall t1', \forall t2, \\ \forall t2', \forall U, \forall \Gamma\end{array} \quad \frac{\Gamma \triangleright 1 : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\frac{\Gamma \triangleright t1 \overset{!j}{\to}_U t1' \quad \Gamma \triangleright t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'}\right)}\right)}{\left(\begin{array}{l}\forall t1, \forall t1', \forall t2, \\ \forall t2', \forall U, \forall \Gamma\end{array} \quad \frac{\Gamma \triangleright 1 : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\frac{\Gamma \triangleright t1 \overset{!j+1}{\to}_U t1' \quad \Gamma \triangleright t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'}\right)}\right)}$$

Choose arbitrary j.

Assume(A): $\begin{array}{l}\forall t1, \forall t1', \forall t2, \\ \forall t2', \forall U, \forall \Gamma\end{array} \quad \frac{\Gamma \triangleright 1 : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\frac{\Gamma \triangleright t1 \overset{!j}{\to}_U t1' \quad \Gamma \triangleright t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'}\right)}$

Derive:

$\begin{array}{l}\forall t1, \forall t1', \forall t2, \\ \forall t2', \forall U, \forall \Gamma\end{array} \quad \frac{\Gamma \triangleright 1 : t1 \to t2 \quad U \not\to \quad C(U)}{\left(\frac{\Gamma \triangleright t1 \overset{!j+1}{\to}_U t1' \quad \Gamma \triangleright t2 \overset{!}{\to}_U t2'}{t1' \approx_U t2'}\right)}$

Choose arbitrary t1, t1', t2, t2', U, and Γ.

Assume: $\Gamma \triangleright 1 : t1 \to t2 \quad U \not\to \quad C(U)$

Assume: $\Gamma \triangleright t1 \overset{!j+1}{\to}_U t1' \quad \Gamma \triangleright t2 \overset{!}{\to}_U t2'$

Derive: $t1' \approx_U t2'$

Because $\Gamma \triangleright t1 \overset{!j+1}{\to}_U t1'$, there exists t3 such that $\Gamma \triangleright t1 \to_U t3$ and $\Gamma \triangleright t3 \overset{!j}{\to}_U t1'$

Case split on the form of $\Gamma \triangleright t1 \to_U t3$

- EMU-BASE: $t1 =_U t4$ and $\Gamma \triangleright t4 \to t3$
  - From lemma 4.2.8.1, $t2 \approx_U t3$
  - From lemma 4.2.3, $\Gamma \triangleright t2 \downarrow_U t3$
  - That is, $\Gamma \triangleright t2 \overset{!}{\to}_U t2'$, $\Gamma \triangleright t3 \overset{!}{\to}_U t1'$, and $t2' \approx_U t1'$
  - From EQ-SYMM, $t1' \approx_U t2'$

- EMU-IND: $t1 = E_1^+[a1]$ where $\Gamma, Bound(E_1^+) \rhd a1 \rightarrow_U a3$
  - So $t3 = E_1^+[a3]$
  - Note that if a1 is inactive, then a3 is the same sort of inactive term (from lemma 4.2.8.4)
  - So because $\Gamma \rhd 1 : t1 \rightarrow t2$, there exists t3' such that $\Gamma \rhd 1 : t3 \rightarrow t3'$
  - Choose arbitrary t3' such that $\Gamma \rhd 1 : t3 \rightarrow t3'$
  - Choose arbitrary t3'' such that $\Gamma \rhd t3' \overset{!}{\rightarrow}_U t3''$
  - From A, $\Gamma \rhd t3'' \downarrow_U t1'$
  - From 4.2.7.2, $\Gamma \rhd t3'' \downarrow_U t2'$
  - From lemma 4.2.8.7, $\Gamma \rhd t1' \downarrow_U t2'$
  - Because $N(U, \rightarrow)$, $\Gamma \rhd N(t1', \rightarrow_U)$, and $\Gamma \rhd N(t2', \rightarrow_U)$, $\Gamma \rhd t1' \downarrow_U t2'$ implies $t1' \approx_U t2'$
  - So $t1' \approx_U t2'$

Lemma 4.2.7.2:

$$\forall E_1^+, \forall a1, \forall a1', \forall t2, \forall t2', \forall U, \forall \Gamma \quad \frac{\Gamma \rhd 1 : E_1^+[a1] \rightarrow t2 \quad U \not\rightarrow \quad C(U) \qquad \left(\frac{\Gamma \rhd E_1^+[a1] \rightarrow_U E_1^+[a1'] \quad \Gamma \rhd E_1^+[a1'] \rightarrow t2'}{\Gamma \rhd t2 \downarrow_U t2'} \right)}{}$$

Choose arbitrary E_1^+, a1, a1', t2, t2', U, and Γ.

Assume: $\Gamma \rhd 1 : E_1^+[a1] \rightarrow t2 \quad U \not\rightarrow \quad C(U)$

Assume: $\Gamma \rhd E_1^+[a1] \rightarrow_U E_1^+[a1'] \quad \Gamma \rhd E_1^+[a1'] \rightarrow t2'$

Derive: $\Gamma \rhd t2 \downarrow_U t2'$

Case split on the form of $\Gamma \rhd E_1^+[a1] \rightarrow^1 t2$

- Case $E_1^+[a1]$ = (fun r(x:!):!.b I)
  - So t2 = b[I/x][fun r(x:!):!.b/r]
  - Case split on the form of $E_1^+[a1]$
    - Case a1 = fun r(x:!):!.b
      - So a1' = fun r(x:!):!.b' where $\Gamma, x \rhd b \rightarrow_U b'$
      - So t2' = b'[I/x][fun r(x:!):!.b'/r]
      - From lemma 4.2.7.5, $\Gamma \rhd t2 \downarrow_U t2'$
    - Case a1 = I
      - From lemma 4.2.8.4 a1' = I' where I' is an inactive term
      - Then $\Gamma \rhd I \rightarrow_U I'$
      - Then t2' = b[I'/x][fun r(x:!):!.b/r]
      - From lemma 4.2.7.4, $\Gamma \rhd t2 \downarrow_U t2'$

- Case $E_1^+[a1] = match\ c_n \bar{d}\ with\ c_1 \bar{x}_1 => b_1 \mid ... \mid c_n \bar{x}_n => b_n \mid ...end$
  - So t2 = $b_n[\bar{d} / \bar{x}_n]$
  - Case split on the form of $E_1^+[a1]$
    - Case a1 = $(c_n\ \bar{d})$
      - Because c_n has no holes, and from lemma 4.2.8.3 $\Gamma \triangleright c_n \not\rightarrow_U$, then $\exists d_n \in \bar{d}$, $\Gamma \triangleright d_n \rightarrow_U e_n$
      - So let $\bar{e} = d_{1...n-1}, e_n, d_{n+1...}$
      - So $\Gamma \triangleright \bar{d} \downarrow_U \bar{e}$
      - So a1' = $(c_n\ \bar{e})$
      - Then t2'= $b_n[\bar{e} / \bar{x}_n]$
      - From lemma 4.2.7.4, $\Gamma \triangleright t2 \downarrow_U t2'$
    - Case a1 = b_n
      - So $\Gamma, \bar{x} \triangleright b_n \rightarrow_U b_n'$
      - So t2' = $b_n'[\bar{d} / \bar{x}_n]$
      - From lemma 4.2.7.5, $\Gamma \triangleright t2 \downarrow_U t2'$
    - Case a1 = b_m where $m \neq n$
      - Then t2' is $b_n[\bar{d} / \bar{x}_n]$
      - Because t2=t2', $\Gamma \triangleright t2 \downarrow_U t2'$ from lemma 4.2.3
- Case $E_1^+[a1] = let\ x = I\ in\ b$
  - So t2 = b[I/x]
  - Case split on the form of $E_1^+[a1]$
    - Case a1 = I
      - Then $\Gamma \triangleright I \rightarrow_U I'$
      - Then t2'=b[I'/x]
      - From lemma 4.2.7.4, $\Gamma \triangleright t2 \downarrow_U t2'$
    - Case a1 = b
      - Then $\Gamma, x \triangleright b \rightarrow_U b'$
      - Then t2'=b'[I/x]
      - From lemma 4.2.7.5, $\Gamma \triangleright t2 \downarrow_U t2'$

Lemma 4.2.7.3:

$$\forall a1, \forall a2, \forall U, \forall \Gamma, \forall E^+ \quad \frac{\Gamma, Bound(E^+) \triangleright a1 \downarrow_U a2}{\Gamma \triangleright E^+[a1] \downarrow_U E^+[a2]}$$

(Note E^+ is the arbitrary depth context)

Choose arbitrary a1, a2, U, E^+, and Γ.
Assume: $\Gamma, Bound(E^+) \triangleright a1 \downarrow_U a2$
Derive: $\Gamma \triangleright E^+[a1] \downarrow_U E^+[a2]$

Proof by induction on the structure of E^+

- Base case: E^+ is *
  - Then Bound(E^+) = { }
  - Then $\Gamma, Bound(E^+) \triangleright a1 \downarrow_U a2$ implies $\Gamma \triangleright E^+[a1] \downarrow_U E^+[a2]$
  - So $\Gamma \triangleright E^+[a1] \downarrow_U E^+[a2]$
- Step case: $E^+ = E^+'[E_1^+]$
  - So $\Gamma, Bound(E^+'), Bound(E_1^+) \triangleright a1 \downarrow_U a2$
  - From lemma 4.2.2.6, $\Gamma, Bound(E^+') \triangleright E_1^+[a1] \downarrow_U E_1^+[a2]$
  - From I.H., $\Gamma \triangleright E^+'[E_1^+[a1]] \downarrow_U E^+'[E_1^+[a2]]$
  - That is, $\Gamma \triangleright E^+[a1] \downarrow_U E^+[a2]$

Lemma 4.2.7.4:

$$\forall \overline{a}, \forall \overline{a}', \forall b, \forall \overline{x}, \forall \Gamma \quad \frac{\Gamma \triangleright \overline{a} \downarrow_U \overline{a}'}{\Gamma \triangleright b[\overline{a} / \overline{x}] \downarrow_U b[\overline{a}' / \overline{x}]}$$

Proof by induction on $b[\overline{a} / \overline{x}]$:

- Base case $b = x_n$ and $(Vars(x_n) \cup Vars(a_n)) \cap \Gamma = \varnothing$
  - So $b[\overline{a} / \overline{x}] = a_n$
  - Because $\Gamma \triangleright \overline{a} \downarrow_U \overline{a}'$, $\Gamma \triangleright a_n \downarrow_U a_n'$
  - Because $\Gamma \triangleright a_n \downarrow_U a_n'$ and $(Vars(x_n) \cup Vars(a_n)) \cap \Gamma = \varnothing$, $(Vars(x_n) \cup Vars(a_n')) \cap \Gamma = \varnothing$
  - So $b[\overline{a}' / \overline{x}] = a_n'$
  - Because $\Gamma \triangleright a_n \downarrow_U a_n'$, $\Gamma \triangleright b[\overline{a} / \overline{x}] \downarrow_U b[\overline{a}' / \overline{x}]$
- Step case $b = E_1^*[\overline{c}]$ and $\forall n \quad \Gamma, Bound(E_1^*, n) \triangleright k : c_n[\overline{a} / \overline{x}] = d_n$
  - I.H. is $\forall \overline{a}, \forall \overline{a}', \forall b, \forall \overline{x}, \forall \Gamma \quad \frac{\Gamma \triangleright \overline{a} \downarrow_U \overline{a}'}{\Gamma \triangleright k : b[\overline{a} / \overline{x}] \downarrow_U b[\overline{a}' / \overline{x}]}$
  - So $b[\overline{a} / \overline{x}] = E_1^*[\overline{d}]$
  - Let $\overline{d}' = \overline{c}[\overline{a}' / \overline{x}]$
  - Then $b[\overline{a}' / \overline{x}] = E_1^*[\overline{d}']$ (Note: even if $b = x_n$, CAS-BASE cannot apply because $(Vars(x_n) \cup Vars(a_n)) \cap \Gamma \neq \varnothing$)
  - From I.H., $\forall n \quad \Gamma, Bound(E_1^*, n) \triangleright d_n \downarrow_U d_n'$

- From lemma 4.2.2.6, $\Gamma \triangleright E_1^*[\overline{d}] \downarrow_U E_1^*[\overline{d}']$
- That is, $\Gamma \triangleright b[\overline{a}/\overline{x}] \downarrow_U b[\overline{a}'/\overline{x}]$

Lemma 4.2.7.5:

$$\forall \overline{a}, \forall b, \forall b', \forall \overline{x}, \forall \Gamma \quad \frac{\Gamma, \overline{x} \triangleright b \rightarrow_U b'}{\Gamma \triangleright b[\overline{a}/\overline{x}] \downarrow_U b'[\overline{a}/\overline{x}]}$$

Choose arbitrary $\overline{a}, b, b', \overline{x}$, and Γ.
Assume: $\Gamma, \overline{x} \triangleright b \rightarrow_U b'$
Derive: $\Gamma \triangleright b[\overline{a}/\overline{x}] \downarrow_U b'[\overline{a}/\overline{x}]$

Then there exists E^+, c, and c' such that b = E^+[c] and $\Gamma, \overline{x}, Bound(E^+) \triangleright c \rightarrow_U c'$
Note that E^+ is the "deepest" possible context that fits the above description.
So b' = E^+[c']
In terms of E^+[c], $b[\overline{a}/\overline{x}] = (E^+[c])[\overline{a}/\overline{x}]$
Because $\Gamma, \overline{x}, Bound(E^+) \triangleright c \rightarrow_U c'$ and E^+ is the deepest possible context, none of $\overline{x}$ appear in c or c'
So $b[\overline{a}/\overline{x}] = (E^+[c])[\overline{a}/\overline{x}] = (E^+[\overline{a}/\overline{x}])[c]$
By a similar argument, $b'[\overline{a}/\overline{x}] = (E^+[c'])[\overline{a}/\overline{x}] = (E^+[\overline{a}/\overline{x}])[c']$
Substitution cannot affect the "binding" variables in a context, so Bound(E^+) = Bound($E^+[\overline{a}/\overline{x}]$)
Then $\Gamma, x, Bound(E^+[\overline{a}/\overline{x}]) \triangleright c \rightarrow_U c'$
Then $\Gamma, x, Bound(E^+[\overline{a}/\overline{x}]) \triangleright c \downarrow_U c'$
From lemma 4.2.7.3, $\Gamma, \overline{x} \triangleright (E^+[\overline{a}/\overline{x}])[c] \downarrow_U (E^+[\overline{a}/\overline{x}])[c']$
Because neither of the terms above contain any of the variables in $\overline{x}$,
$\Gamma \triangleright (E^+[\overline{a}/\overline{x}])[c] \downarrow_U (E^+[\overline{a}/\overline{x}])[c']$
That is, $\Gamma \triangleright b[\overline{a}/\overline{x}] \downarrow_U b'[\overline{a}/\overline{x}]$

Lemma 4.2.8.1

$$\forall t1, \forall t2, \forall t1', \forall t2' \quad \forall U, \forall \Gamma \qquad \frac{t1 \approx_U t2 \quad \Gamma \triangleright t1 \rightarrow t1' \quad \Gamma \triangleright t2 \rightarrow t2' \quad U \not\rightarrow \quad C(U)}{t1' \approx_U t2'}$$

Proof by induction on $\Gamma \triangleright t1 \rightarrow t1'$

- Case t1= $E_1[a1]$ where $\Gamma \triangleright k : a1 \rightarrow a1'$
  - I.H. is

$$\forall t1, \forall t2, \forall t1', \forall t2' \quad \forall U, \forall \Gamma \qquad \frac{t1 \approx_U t2 \quad \Gamma \triangleright k : t1 \rightarrow t1' \quad \Gamma \triangleright t2 \rightarrow t2' \quad U \not\rightarrow \quad C(U)}{t1' \approx_U t2'}$$

- From lemma 4.2.8.5, t1 = $E_1^*[b_1...b_{n-1}, a1, b_{n+1...Holes(E_1^*)}]$ and t2 = $E_1^*[d_1...d_{n-1}, a2, d_{n+1...Holes(E_1^*)}]$ for some $\overline{b}$ and $\overline{d}$
- Then $a1 \approx_U a2$
- Case split on the form of E_1^*
  - Case t1=(a1 g1) and t2=(a2 g2)
    - So t1' = (a1' g1)
    - Case split on the form of $\Gamma \triangleright t2 \to t2'$
      - Case $\Gamma \triangleright a2 \to a2'$
        - From I.H. $a1' \approx_U a2'$
        - So t2' = (a2' g2)
        - From EQ-CTXT $t1' \approx_U t2'$
      - Case $a2 \in I$ and $\Gamma \triangleright g2 \to g2'$
        - From lemma 4.2.8.3, $\Gamma \triangleright \neg(\exists a1'' =_U a1 \quad a1'' \to)$
        - The previous statement contradicts $\Gamma \triangleright a1 \to a1'$
        - So conclude $t1' \approx_U t2'$
      - Case a2=fun x(a:!):!.t and $g2 \in I$
        - From lemma 4.2.8.3, $\Gamma \triangleright \neg(\exists a1'' =_U a1 \quad a1'' \to)$
        - The previous statement contradicts $\Gamma \triangleright a1 \to a1'$
        - So conclude $t1' \approx_U t2'$
  - Case t1=(f1 a1) and t2=(f2 a2) where $f1 \in I$
    - So t1' = (f1 a1')
    - Case split on the form of $\Gamma \triangleright t2 \to t2'$
      - Case $\Gamma \triangleright f2 \to f2'$
        - From lemma 4.2.8.3, $\Gamma \triangleright \neg(\exists f2'' =_U f2 \quad f2'' \to)$
        - The previous statement contradicts $\Gamma \triangleright f2 \to f2'$
        - So conclude $t1' \approx_U t2'$
      - Case $f2 \in I$ and $\Gamma \triangleright a2 \to a2'$
        - From I.H. $a1' \approx_U a2'$
        - So t2' = (f2 a2')
        - Then $t1' \approx_U t2'$
      - Case f2=fun x(a:!):!.t and $a2 \in I$

- From lemma 4.2.8.3,
  $\Gamma \triangleright \neg(\exists a1'' =_U a1 \quad a1'' \rightarrow)$
- The previous statement contradicts
  $\Gamma \triangleright a1 \rightarrow a1'$
- So conclude $t1' \approx_U t2'$

- Case t1= match a1 with $c_1\bar{x}_1 => s_1 \mid ... \mid c_n\bar{x}_n => s_n$ *end*
  - So t2 = match a2 with $c_1\bar{y}_1 => v_1 \mid ... \mid c_n\bar{y}_n => v_n$ *end*
  - Then t1' = match a1' with $c_1\bar{x}_1 => s_1 \mid ... \mid c_n\bar{x}_n => s_n$ *end*
  - Case split on the form of $\Gamma \triangleright t2 \rightarrow t2'$
    - Case $\Gamma \triangleright a2 \rightarrow a2'$
      - From I.H. $a1' \approx_U a2'$
      - So t2' = match a2' with
        $c_1\bar{y}_1 => v_1 \mid ... \mid c_n\bar{y}_n => v_n$ *end*
      - From EQ-CTXT $t1' \approx_U t2'$
    - Case $a2 = (c_i\bar{I})$
      - From lemma 4.2.8.3,
        $\Gamma \triangleright \neg(\exists a1'' =_U a1 \quad a1'' \rightarrow)$
      - The previous statement contradicts
        $\Gamma \triangleright a1 \rightarrow a1'$
      - So conclude $t1' \approx_U t2'$
- Case t1= let x1 = a1 in b1
  - So t2 = let x2 = a2 in b2
  - Then t1' = let x1 = a1' in b1
  - Case split on the form of $\Gamma \triangleright t2 \rightarrow t2'$
    - Case $\Gamma \triangleright a2 \rightarrow a2'$
      - From I.H. $a1' \approx_U a2'$
      - So t2' = let x2 = a2' in b2
      - From EQ-CTXT $t1' \approx_U t2'$
    - Case $a2 \in I$
      - From lemma 4.2.8.3,
        $\Gamma \triangleright \neg(\exists a1'' =_U a1 \quad a1'' \rightarrow)$
      - The previous statement contradicts
        $\Gamma \triangleright a1 \rightarrow a1'$
      - So conclude $t1' \approx_U t2'$

- Case t1=(F I) where F= fun x(a:!):!.t
  - So t1' = t[I / a] [F/x]
  - From lemma 4.2.8.5, t2=(F' I') and $F \approx_U F'$ and $I \approx_U I'$
  - From lemma 4.2.8.3, $\Gamma \triangleright F' \not\rightarrow$ and $\Gamma \triangleright I' \not\rightarrow$

- Because $\Gamma \triangleright t2 \rightarrow t2'$ and all the sub-terms of t2 don't take a step in $\rightarrow$, F' is a fun term
- Because C(U), and $F \approx_U F'$, F'= fun x'(a':!):!.t' where $t' \approx_U t[a'/a]$
- So t2'=t'[I'/a'][F'/x']
- Because $t' \approx_U t[a'/a]$, $I \approx_U I'$, and $F \approx_U F'$, $t'[I'/a'][F'/x'] \approx_U t[I/a][F/x]$
- From EQ-SYMM, $t[I/a][F/x] \approx_U t'[I'/a'][F'/x']$
- That is, $t1' \approx_U t2'$

- Case t1=match ($c_i \bar{I}$) by x y return ! with
  $c_1 \bar{x}_1 \Rightarrow s_1 \mid ... \mid c_i \bar{x}_i \Rightarrow s_i \mid ... \mid c_n \bar{x}_n \Rightarrow s_n$ *end*
  - So t1'= $s_i [\bar{I} / \bar{x}_i]$
  - From lemma 4.2.8.5, t2= match a by x y return ! with
    $c_1 \bar{x}_1 \Rightarrow s_1' \mid ... \mid c_i \bar{x}_i \Rightarrow s_i' \mid ... \mid c_n \bar{x}_n \Rightarrow s_n'$ *end* where $a \approx_U (c_i \bar{I})$ and $\forall n \quad s_n \approx_U s_n'$
  - Because C(U) and $a \approx_U (c_i \bar{I})$, $\forall a' =_U a$ where $a' = (f\ \bar{t})$, $f = c_i$. From lemma 4.2.8.3, $\forall a'$ where $a' =_U a$, $\Gamma \triangleright a' \not\rightarrow$ a' does not take a step, so none of $\bar{t}$ will take a step in $\rightarrow$. Because $a' =_U a$, $\bar{t} =_U \bar{I}$
  - So t2' = $s_i'[\bar{t} / \bar{x}_i]$
  - Because $\bar{t} =_U \bar{I}$, and $s_i \approx_U s_i'$, $t1' \approx_U t2'$ by EQ-CTXT
- Case t1=let x = I by y in t
  - So t1' = t[I/x]
  - From lemma 4.2.8.5, t2=let x' = I' by y' in t' where $I \approx_U I'$ and $t[x'/x] \approx_U t'$
  - So t2' = t'[I'/x']
  - So $t1' \approx_U t2'$ by EQ-CTXT
- Case t1=E[abort !]
  - So t1' = abort !
  - Then t2 = E[abort !]
  - So t2' = abort !
  - So $t1' \approx_U t2'$

<u>Lemma 4.2.8.2</u>

$$\forall t1, \forall t2, \forall U, \forall \bar{a}, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad t1 = E_1^*[\bar{a}]}{\exists \bar{b} \quad t2 =_U E_1^*[\bar{b}]}$$

Proof by induction on $t1 \approx_U t2$

- Case t1 = t2
  - Let $\bar{b} = \bar{a}$

- Then t2 = $E_1^*[\bar{a}]$

- Case $\{t1 = t2\} \in U$
  - Let $\bar{b} = \bar{a}$
  - Then $t2 =_U E_1^*[\bar{a}]$
- Case $k : t2 \approx_U t1$
  - I.H. is $\forall t1, \forall t2, \forall U, \forall \bar{a}, \forall \Gamma \quad \dfrac{k : t1 \approx_U t2 \quad t1 = E_1^*[\bar{a}]}{\exists \bar{b} \quad t2 =_U E_1^*[\bar{b}]}$
  - From I.H., $\exists \bar{b} \quad t2 =_U E_1^*[\bar{b}]$
- Case $\exists t3 \quad k : t1 \approx_U t3 \quad k : t3 \approx_U t2$
  - I.H. is $\forall t1, \forall t2, \forall U, \forall \bar{a}, \forall \Gamma \quad \dfrac{k : t1 \approx_U t2 \quad t1 = E_1^*[\bar{a}]}{\exists \bar{b} \quad t2 =_U E_1^*[\bar{b}]}$
  - From I.H., $\exists \bar{b} \quad t3 =_U E_1^*[\bar{b}]$
  - Choose arbitrary $E_1^*[\bar{b}]$ where $t3 =_U E_1^*[\bar{b}]$
  - From I.H., $\exists \bar{b}' \quad t2 =_U E_1^*[\bar{b}']$
- Case $t1 = E_1^+[a1] \quad t2 = E_1^+[a2] \quad k : a1 \approx_U a2$
  - So t1 = $E_1^*[\bar{c}, a1, \bar{d}]$ and t2 = $E_1^*[\bar{c}, a2, \bar{d}]$
  - So let $\bar{b}$ be $\bar{c}, a2, \bar{d}$
  - Then $t2 =_U E_1^*[\bar{b}]$

Lemma 4.2.8.2.1

$$\forall t1, \forall t2, \forall U, \forall \bar{a}, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad t1 = (c\ \bar{a})}{\exists \bar{b} \quad t2 =_U (c\ \bar{b})}$$

From lemma 4.2.8.2, $\exists d, \bar{d}$ such that $t2 =_U (d\ \bar{d})$
So $d \approx_U c$
From lemma 4.2.8.2, $d =_U c$
So $t2 =_U (c\ \bar{d})$

Lemma 4.2.8.2.2

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{t1 \approx_U t2 \quad t1 \in I}{\exists t2' \in I \quad t2' =_U t2}$$

Note I is the set of inactive terms

t1 = $E_1^*[\bar{a}]$ for some $\bar{a}$
From lemma 4.2.8.2, $\exists \bar{b} \quad t2 =_U E_1^*[\bar{b}]$
Proof by induction on $E_1^*[\bar{a}]$

- Case t1 = (C $\bar{a}$) where C is a constant

- I.H. is $\forall n, \forall b, \forall U, \forall \Gamma \quad \dfrac{a_n \approx_U b \quad a_n \in I}{\exists b' \in I \quad b' =_U b}$
- From lemma 4.2.8.2.1, $\exists \bar{b} \quad t2 =_U (C\ \bar{b})$
- From I.H. $\forall n \quad \exists b_n' \in I \quad b_n' =_u b_n$
- So $(C\ \bar{b}') \in I$, let $t2' = (C\ \bar{b}')$

- Case otherwise
  - All other inactive E_1^* will be inactive regardless of their subterms
  - So $\exists t2' \in I \quad t2' =_U t2$ from lemma 4.2.8.2

Lemma 4.2.8.3

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{U \not\to \quad C(U) \quad t1 \approx_U t2 \quad t1 \in I}{\forall t2' =_U t2 \quad \Gamma \rhd t2' \not\to}$$

Note I is the set of inactive terms

Choose arbitrary t2' such that $t2' =_U t2$

(Proof by contradiction)

Assume $\Gamma \rhd t2' \to$

Because $t2' \approx_U t1$ and lemma 4.2.8.5, $t1 = E_1^*[\bar{a}]$ and $t2' = E_1^*[\bar{b}]$ for some $\bar{a}, \bar{b}$

Proof by induction on $E_1^*[\bar{a}]$

- Case t1 = (c1 a1) where c1 is a constant
  - So t2' = (f2 a2)
  - Case split on the form of $\Gamma \rhd t2' \to$
    - Case $\Gamma \rhd f2 \to$
      - From I.H. $\Gamma \rhd f2 \not\to$
      - Contradiction, conclude $\Gamma \rhd t2' \not\to$
    - Case $f2 \in I$ and $a2 \to$
      - From I.H. $\Gamma \rhd a2 \not\to$
      - Contradiction, conclude $\Gamma \rhd t2' \not\to$
    - Case f2= fun x(y:!):!.d and $a2 \in I$
      - Contradicts C(U) because c1 $\approx_U$ fun x(y:!):!.d
      - So conclude $\Gamma \rhd t2' \not\to$
- Case otherwise
  - All other inactive E_1^* will be inactive regardless of their subterms
  - So $t2' \in I$
  - Therefore $\Gamma \rhd t2' \not\to$

In all cases, we get something that contradicts $\Gamma \triangleright t2' \rightarrow$
So $\Gamma \triangleright t2' \not\rightarrow$

Lemma 4.2.8.4:

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{U \not\rightarrow \quad C(U) \quad \Gamma \triangleright t1 \rightarrow_U t2 \quad t1 \in I}{t2 \in I}$$

Induction on $\Gamma \triangleright t1 \rightarrow_U t2$

- Case $t1 = E_1^+[a1]$, $\Gamma, Bound(E_1^+) \triangleright k : a1 \rightarrow_U a2$, $t2 = E_1^+[a2]$
  - I.H. is $\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{U \not\rightarrow \quad C(U) \quad \Gamma \triangleright k : t1 \rightarrow_U t2 \quad t1 \in I}{t2 \in I}$
  - Case split on the form of $E_1^+[a1]$
    - Case $E_1^+[a1] = (c\ \bar{I})$ where c is a constant and $\bar{I}$ is a list of inactive terms
      - From lemma 4.2.8.3, and because c has no holes, $c \not\rightarrow_U$
      - So some $I_n \in \bar{I}$ is a1
      - From I.H., $a2 \in \bar{I}$
      - So $t2 = (c\ \bar{I}')$ where $\bar{I}'$ is a list of inactive terms
      - So $t2 \in I$
    - Case otherwise
      - All other E_1^+ are inactive regardless of the subterms
      - So $t2 \in I$
- Case $t1 =_U t3$ and $\Gamma \triangleright t3 \rightarrow t2$
  - From lemma 4.2.8.3, this t3 cannot exist
  - This case results in a contradiction, conclude $t2 \in I$

Lemma 4.2.8.5

$$\forall t1, \forall t2, \forall U, \forall \Gamma \quad \frac{U \not\rightarrow \quad t1 \approx_U t2 \quad \Gamma \triangleright t1 \rightarrow_U}{\exists E_1^*, \bar{b}, \bar{b}' \quad t1 = E_1^*[\bar{b}] \quad t2 = E_1^*[\bar{b}']}$$

Proof by induction on $t1 \approx_U t2$:

- Case t1 = t2
  - Then trivially, $\exists E_1^*, \bar{b}, \bar{b}' \quad t1 = E_1^*[\bar{b}] \quad t2 = E_1^*[\bar{b}']$
- Case $\{t1 = t2\} \in U$
  - Contradicts $U \not\rightarrow$
  - So conclude $\exists E_1^*, \bar{b}, \bar{b}' \quad t1 = E_1^*[\bar{b}] \quad t2 = E_1^*[\bar{b}']$
- Case $k : t2 \approx_U t1$

- I.H. is $\forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{N(U,\rightarrow) \quad k:t1 \approx_U t2 \quad \Gamma \rhd t1 \rightarrow_U}{\exists E_1^*, \bar{b}, \bar{b}' \quad t1 = E_1^*[\bar{b}] \quad t2 = E_1^*[\bar{b}']}$
- From I.H. $\exists E_1^*, \bar{b}, \bar{b}' \quad t1 = E_1^*[\bar{b}] \quad t2 = E_1^*[\bar{b}']$

- Case $\exists t3 \quad k:t1 \approx_U t3 \quad k:t3 \approx_U t2$
  - I.H. is $\forall t1, \forall t2, \forall U, \forall \Gamma \quad \dfrac{N(U,\rightarrow) \quad k:t1 \approx_U t2 \quad \Gamma \rhd t1 \rightarrow_U}{\exists E_1^*, \bar{b}, \bar{b}' \quad t1 = E_1^*[\bar{b}] \quad t2 = E_1^*[\bar{b}']}$
  - From I.H. $\exists E_1^*, \bar{b}, \bar{b}' \quad t1 = E_1^*[\bar{b}] \quad t3 = E_1^*[\bar{b}']$
  - From I.H. $\exists E_1^{*}', \bar{b}'', \bar{b}''' \quad t3 = E_1^{*}'[\bar{b}''] \quad t2 = E_1^{*}'[\bar{b}''']$
  - Because $t3 = E_1^*[\bar{b}']$ and $t3 = E_1^{*}'[\bar{b}'']$, $E_1^* = E_1^{*}'$
  - Let $\bar{b}' = \bar{b}'''$
  - So $\exists E_1^*, \bar{b}, \bar{b}' \quad t1 = E_1^*[\bar{b}] \quad t2 = E_1^*[\bar{b}']$
- Case $t1 = E_1^+[a1] \quad t2 = E_1^+[a2] \quad a1 \approx_U a2$
  - So t1 = $E_1^*[\bar{b}, a1, \bar{d}]$ and t2 = $E_1^*[\bar{b}, a2, \bar{d}]$
  - So $\exists E_1^*, \bar{b}, \bar{b}' \quad t1 = E_1^*[\bar{b}] \quad t2 = E_1^*[\bar{b}']$

Lemma 4.2.8.6:

$$\forall t1, \forall t1', \forall t2, \forall U, \forall \Gamma \quad \frac{U \not\rightarrow \quad C(U) \quad t1 \approx_U t2 \quad t1 =_U t1' \quad \Gamma \rhd t1' \rightarrow}{\exists t2' =_U t2 \quad \Gamma \rhd t2' \rightarrow}$$

Proof by induction on $t1 \approx_U t2$:

- Case t1 = t2
  - So $t1 =_U t2$ and from transitivity, $t1' =_U t2$
  - Let t2' = t1'
  - Then $\exists t2' =_U t2 \quad \Gamma \rhd t2' \rightarrow$
- Case $\{t1 = t2\} \in U$
  - So $t1 =_U t2$ and from transitivity, $t1' =_U t2$
  - Let t2' = t1'
  - Then $\exists t2' =_U t2 \quad \Gamma \rhd t2' \rightarrow$
- Case $k:t2 \approx_U t1$
  - I.H. is $\forall t1, \forall t1', \forall t2, \forall U, \forall \Gamma \quad \dfrac{U \not\rightarrow \quad C(U) \quad k:t1 \approx_U t2 \quad t1 =_U t1' \quad \Gamma \rhd t1' \rightarrow}{\exists t2' =_U t2 \quad \Gamma \rhd t2' \rightarrow}$
  - From I.H. $\exists t2' =_U t2 \quad \Gamma \rhd t2' \rightarrow$
- Case $\exists t3 \quad k:t1 \approx_U t3 \quad k:t3 \approx_U t2$
  - I.H. is $\forall t1, \forall t1', \forall t2, \forall U, \forall \Gamma \quad \dfrac{U \not\rightarrow \quad C(U) \quad k:t1 \approx_U t2 \quad t1 =_U t1' \quad \Gamma \rhd t1' \rightarrow}{\exists t2' =_U t2 \quad \Gamma \rhd t2' \rightarrow}$

- From I.H. $\exists t3' =_U t3 \quad \Gamma \triangleright t3' \rightarrow$
- Choose arbitrary t3' such that $t3' =_U t3$ and $\Gamma \triangleright t3' \rightarrow$
- From I.H. $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$

- Case $t1 = E_1^+[a1] \quad t2 = E_1^+[a2] \quad k : a1 \approx_U a2$
  - I.H. is

$$\forall t1, \forall t1', \forall t2, \forall U, \forall \Gamma \quad \frac{U \not\rightarrow \quad C(U) \quad k : t1 \approx_U t2 \quad t1 =_U t1' \quad \Gamma \triangleright t1' \rightarrow}{\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow}$$

  - Case split on the form of t1
    - Case t1 = (a1 b)
      - Then t2 = (a2 b)
      - From lemma 4.2.8.5, t1' = (a1'' b1'')
      - Case split on the form of $\Gamma \triangleright t1' \rightarrow$
        - Case $\Gamma \triangleright a1'' \rightarrow a1'$
          - From I.H. $\exists a2'' =_U a2 \quad \Gamma \triangleright a2'' \rightarrow$
          - So, $t2 =_U (a2'' b)$
          - Let t2' = $(a2'' b)$
          - So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$
        - Case $a1'' \in I$ and $\Gamma \triangleright b1'' \rightarrow b1'$
          - From lemma 4.2.8.2.2, $\exists a2'' \in I \quad a2'' =_U a2$
          - So $t2 =_U (a2'' b1'')$
          - Let t2' = $(a2'' b1'')$
          - So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$
        - Case a1''=fun r(x:!):!.y and $b1'' \in I$
          - From lemma 4.2.8.2, $\exists a2'' =_U a2$ where a2''=fun r(x':!):!.y'
          - So $t2 =_U (a2'' b1'')$
          - Let t2' = $(a2'' b1'')$
          - So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$
    - Case t1 = (f a1)
      - Then t2 = (f a2)
      - From lemma 4.2.8.5, t1' = (f1'' a2'')
      - Case split on the form of $\Gamma \triangleright t1' \rightarrow$
        - Case $\Gamma \triangleright f1'' \rightarrow f'$
          - So $t2 =_U (f1'' a2)$
          - Let t2' = $(f1'' a2)$
          - So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$
        - Case $f1'' \in I$ and $a1'' \rightarrow a1'$

- From I.H. $\exists a2" =_U a2 \quad \Gamma \triangleright a2" \rightarrow$
- So $t2 =_U (f1'' \; a2'')$
- Let t2' = $(f1" \; a2")$
- So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$

○ Case f1''=fun r(x:!):!.y and $a1'' \in I$

- From lemma 4.2.8.2.2, $\exists a2" \in I \quad a2" =_U a2$
- So $t2 =_U (f1'' \; a2'')$
- Let t2' = $(f1" \; a2")$
- So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$

▪ Case t1 = fun r(x:!):!.a1

- From lemma 4.2.8.5, t1' = fun r(x:!):!.a1'' for some a1''
- Then $\Gamma \triangleright t1' \not\rightarrow$
- The above conclusion contradicts $\Gamma \triangleright t1' \rightarrow$
- Conclude $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$

▪ Case t1 = let x = a1 by y in b

- Then t2 = let x = a2 by y in b
- From lemma 4.2.8.5, t1' = let x = a1'' by y in b1'' for some a1'' and b1''
- Case split on the form of $\Gamma \triangleright t1' \rightarrow$
  - Case $\Gamma \triangleright a1" \rightarrow a1'$
    - From I.H. $\exists a2" =_U a2 \quad \Gamma \triangleright a2" \rightarrow$
    - So $t2 =_U$ let x = a2'' by y in b
    - Let t2' = let x = a2'' by y in b
    - So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$
  - Case $a1'' \in I$
    - From lemma 4.2.8.2, $\exists a2" \in I \quad a2" =_U a2$
    - So $t2 =_U$ let x = a2'' by y in b
    - Let t2' = let x = a2'' by y in b
    - So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$

▪ Case t1 = let x = v by y in a1

- Then t2 = let x = v by y in a2
- From lemma 4.2.8.5, t1' = let x = v1'' by y in a1'' for some v1'' and a1''
- Case split on the form of $\Gamma \triangleright t1' \rightarrow$
  - Case $\Gamma \triangleright v1" \rightarrow v1'$
    - So $t2 =_U$ let x = v1'' by y in a2
    - Let t2' = let x = v1'' by y in a2
    - So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$

        - Case $v1'' \in I$
            - So $t2 =_U$ let x = v1'' by y in a2
            - Let t2' = let x = v1'' by y in a2
            - So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$
- Case t1 = match a1 by x y return ! with $c_1\bar{x}_1$ =>b1|…|$c_n\bar{x}_n$ => bn|…end
    - Then t2 = match a2 by x y return ! with $c_1\bar{x}_1$ =>b1|…|$c_n\bar{x}_n$ => bn|…end
    - From lemma 4.2.8.5, t1' = match a1'' by x y return ! with $c_1\bar{x}_1$ =>b1''|…|$c_n\bar{x}_n$ => bn''|…end for some a1'' and b1''…bn''
    - Case split on the form of $\Gamma \triangleright t1' \rightarrow$
        - Case $\Gamma \triangleright a1'' \rightarrow a1'$
            - From I.H. $\exists a2'' =_U a2 \quad \Gamma \triangleright a2'' \rightarrow$
            - So $t2 =_U$ match a2'' by x y return ! with $c_1\bar{x}_1$ =>b1''|…|$c_n\bar{x}_n$ => bn''|…end
            - Let t2' = match a2'' by x y return ! with $c_1\bar{x}_1$ =>b1''|…|$c_n\bar{x}_n$ => bn''|…end
            - So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$
        - Case a1'' = $c_i\bar{x}$
            - From lemma 4.2.8.2.1, $\exists a2'' =_U a2$ where a2'' = $c_i\bar{x}'$
            - So $t2 =_U$ match a2'' by x y return ! with $c_1\bar{x}_1$ =>b1''|…|$c_n\bar{x}_n$ => bn''|…end
            - Let t2' = match a2'' by x y return ! with $c_1\bar{x}_1$ =>b1''|…|$c_n\bar{x}_n$ => bn''|…end
            - So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$
- Case t1 = match s by x y return ! with $c_1\bar{x}_1$ =>b1|…|$c_i\bar{x}_i$ =>a1|…|$c_n\bar{x}_n$ => bn|…end
    - Then t2 = match s by x y return ! with $c_1\bar{x}_1$ =>b1|…|$c_i\bar{x}_i$ =>a2|…|$c_n\bar{x}_n$ => bn|…end
    - From lemma 4.2.8.5, t1' = match s1'' by x y return ! with $c_1\bar{x}_1$ =>b1''|…|$c_i\bar{x}_i$ =>a2''|…|$c_n\bar{x}_n$ => bn''|…end for some s1'', a1'' and b1''…bn''
    - Case split on the form of $\Gamma \triangleright t1' \rightarrow$
        - Case $\Gamma \triangleright s1'' \rightarrow s1'$
            - So $t2 =_U$ match s1'' by x y return ! with $c_1\bar{x}_1$ =>b1|…|$c_i\bar{x}_i$ =>a2|…|$c_n\bar{x}_n$ => bn|…end

- Let t2' = match s1'' by x y return ! with
  $c_1\bar{x}_1$ =>b1|…|$c_i\bar{x}_i$ =>a2|…|$c_n\bar{x}_n$ => bn|…end
- So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$

○ Case s1'' = $c_i\bar{x}$

- So $t2 =_U$ match s1'' by x y return ! with
  $c_1\bar{x}_1$ =>b1|…|$c_i\bar{x}_i$ =>a2|…|$c_n\bar{x}_n$ => bn|…end
- Let t2' = match s1'' by x y return ! with
  $c_1\bar{x}_1$ =>b1|…|$c_i\bar{x}_i$ =>a2|…|$c_n\bar{x}_n$ => bn|…end
- So $\exists t2' =_U t2 \quad \Gamma \triangleright t2' \rightarrow$

Lemma 4.2.8.7

$$\forall t1, \forall t2, \forall t3, \forall U, \forall \Gamma \quad \frac{\Gamma \triangleright t1 \downarrow_U t2 \quad \Gamma \triangleright t2 \downarrow_U t3 \quad U \not\rightarrow}{\Gamma \triangleright t1 \downarrow_U t3}$$

Choose arbitrary t1, t2, t3, U, and Γ.
Assume: $\Gamma \triangleright t1 \downarrow_U t2 \quad \Gamma \triangleright t2 \downarrow_U t3$
Derive: $\Gamma \triangleright t1 \downarrow_U t3$

Because $\Gamma \triangleright t1 \downarrow_U t2$ and $U \not\rightarrow$, $\forall t1', \forall t2' \quad \Gamma \triangleright t1 \overset{!}{\rightarrow}_U t1' \quad \Gamma \triangleright t2 \overset{!}{\rightarrow}_U t2' \quad t1' \approx_U t2'$

Because $\Gamma \triangleright t2 \downarrow_U t3$ and $U \not\rightarrow$, $\forall t2'', \forall t3'' \quad \Gamma \triangleright t2 \overset{!}{\rightarrow}_U t2'' \quad \Gamma \triangleright t3 \overset{!}{\rightarrow}_U t3'' \quad t2'' \approx_U t3''$

Choose arbitrary t1', t2', t2'', and t3'' (termination is assumed)
From lemma 4.2.3, $t2' \approx_U t2''$
From transitivity, $t1' \approx_U t3''$
So $\Gamma \triangleright t1 \downarrow_U t3$

5 Conclusion and Future Work

This paper presented an algorithm that can be used to deduce the equality of terms in many common cases in OpTT. This algorithm will significantly reduce the amount of effort required to develop most formal proofs in OpTT. However, there a few areas in which additional research would improve the algorithm even further. These areas are discussed in this section.

5.1 Termination

The algorithm in this paper is conjectured to terminate if all terms encountered are terminating with respect to the operational semantics of OpTT. A proof of this conjecture and an investigation into the conditions under which the hypjoin algorithm terminates would be incredibly useful. This information would allow the programmer to fully understand and predict when the algorithm will terminate.

5.2 Clash/Contra

It would be fairly trivial to extend hypjoin so that it is also complete with respect to the clash and contra proof rules which are used to derive conclusions in the presence of contradictions. The algorithm presented in this paper will terminate and fail if inconsistency is detected. It would be just as easy to successfully equate any two terms if the provided equations result in some contradiction.

5.3 Injectivity

Another simple extension involves completeness with respect to the inj proof rule which is used for reasoning about injectivity. The algorithm could simply detect equations that lead to some conclusion that can be derived via injectivity, then that conclusion could be added to the list of equations. For example, if the user-provided equations contain the equation

{(C a) = (C b)} where C is a term constructor, then the algorithm would add {a = b} to the list of user equations.

References

[1] Franz Baader and Tobias Nipkow. "Term Rewriting and All That." Cambridge University Press, Cambridge, 1998

[2] Leo Bachmair and Ashish Tiwari. "Abstract Congruence Closure and Specializations." Proceedings of the 17th International Conference on Automated Deduction, pages 64-78, 2000.

[3] Frédéric Blanqui, Jean-Pierre Jouannaud and Pierre-Yves Strub. "From Formal Proofs to Mathematical Proofs: A safe, incremental way for building first-order decision procedures." http://www.loria.fr/~blanqui/papers/ccic.pdf

[4] Frédéric Blanqui, Jean-Pierre Jouannaud and Pierre-Yves Strub. "Building Decision Procedures in the Calculus of Inductive Constructions." Lecture Notes in Computer Science, pages 328-342, Volume 4646, 2007

[5] Nicolaas Govert de Bruijn. "Lambda Calculus Notation with Nameless Dummies: A Tool for Automatic Formula Manipulation, with Application to the Church-Rosser Theorem." Indagationes Mathematicae, pages 381-392, Volume 34, 1972.

[6] Solomon Feferman. "Predicativity." The Oxford Handbook of Philosophy of Mathematics and Logic, pages 590-624, Oxford University Press, Oxford, 2005.

[7] Donald Knuth and Peter Bendix. "Simple Word Problems in Universal Algebras." Computational Problems in Abstract Algebra, pages 263-297. Pergamon Press, 1970

[8] Aaron Stump and Edwin Westbrook. "Partial Functions in Operational Type Theory." http://www.guru-lang.org/guru.pdf

Vita

Adam Petcher

Date of Birth June 2, 1980

Place of Birth Dallas, Texas

Degrees B.S. Cum Laude, Computer Science, December 2002
Texas Christian University, Fort Worth, Texas

M.S. Computer Science, May 2008
Washington University in Saint Louis, Saint Louis, Missouri

May 2008

Short Title: Joining Mod Gnd Equations in OpTT Adam Petcher, M.S. 2008