並行論理型言語の意味領域としてのガード付きストリーム：意味論的および構文的同値関係について

カサブランカ・ファビオ *, 村上昌己 **, 結縁祥治 *, 稲垣康善 *
* 名古屋大学工学部情報工学科,
** 岡山大学工学部情報工学科

概要

今までに述語記号のラベルを持つガード付きストリーム（I／O ヒストリーズ）が並列論理型プログラミング言語に形式的意味論を与えるのに用いられてきた [Mu90]。しかし、ガード付きストリームの同値関係については何の議論も行なわれていない。本稿では、意味論的同値関係と構文論的同値関係を導入し、さらに両者がすべての有意な場合で一致することを証明する。ガード付きストリームの同値類から得られる領域が意味論の充分な抽象性を議論する上で適切な枠組になる。

Guarded Streams as a Domain for Concurrent Logic Computations: Semantic and Syntactic Equivalence

Fabio CASABLANCA*, Masaki MURAKAMI**, Shōji YUEN*, Yasuyoshi INAGAKI*
* Nagoya University, Department of Information Engineering,
** Okayama University, Department of Information Technology

Abstract

Guarded streams labeled by predicate symbols(I/O histories) have been used to give a formal semantics to concurrent logic languages [Mu90].
However, no results were given on equivalence of guarded streams. This report introduces a semantic and syntactic equivalence. Also, it proves that they coincide in all the situations of interest. The resulting domain of guarded streams'equivalence classes becomes an appropriate framework to discuss full-abstractness of the semantics.

1 Introduction

Concurrent logic languages describe concurrency by means of the logic programming paradigm. A well-known subset is constituted by the family of committed-choice languages (from now on *CLP-languages*) which include, among others, Concurrent Prolog, PARLOG, GHC, Fleng and Oc.

In the last 10 years, various semantics have been proposed for CLP-languages and more recently general frameworks for semantics of concurrent constraint programming languages (*cc-languages*) have been proposed. In all the approaches, special attention has being given to compositionality and full-abstractness.

Domains which have been used to denote agents compositionally include reactive trees[GaLe92], reactive sequences [BoPa90a] (which may be considered to subsume I/O sequences, as used by [GKLS88, GMS89]) bounded trace operators ([SPR91]) and guarded streams[Mu90].

Guarded streams labeled by predicate symbols (I/O histories) have been used to give a formal semantics to perpetual CLP computations [Mu90]. A guarded stream is a set of elementary computational units, *c-units* (in [Mu90], guarded unifications); a c-unit $< \bar{I}|O >$ represents the production of a binding equation, O, prefixed by an equation set $\bar{I}$, which should be satisfied to produce the equation. Exceptional situations which arise with deadlock and failure are expressed assigning to O the special symbols Δ and Φ respectively.

Semantic dependency between c-units is expressed by a partial order. Then guarded streams are c-unit sets over which some conditions are imposed to ensure that the partial order is defined without ambiguity.

We think that I/O histories present,w.r.t. to other models, the following advantages:

- they can be interpreted as guarded logical atoms and sometimes it becomes possible to extend techniques used in logic programming to CLP-programming (an example is given by the extension of guarded streams which deals with OR-compositionality of programs [KaMu93]);

- guarded streams are from start more abstract semantic objects than reactive sequences and bto's and cumbersome saturation conditions can be avoided.

Guarded streams present the advantages pointed out in the previous paragraph, no direct results were given in [Mu90] on full-abstractness. For this purpose, a key point is to be able to decide if two guarded streams are equivalent, because a domain which distinguishes equivalent guarded streams can not be fully-abstract.

In this paper, we discuss suitable notions of meaning and equivalence relation for guarded streams, using the corresponding definition for equation sets. The corresponding domain of guarded streams'equivalence classes becomes an adequate framework to discuss full-abstractness.

Together with the semantic equivalence we have defined a a syntactic equivalence, based on the recognition of sets of "interchangeable" variables. We prove that semantic and syntactic equivalence coincide in all the situations of interest. As it should be expected, the syntactic equivalence is simpler to manipulate and it proves to be a valuable tool when working with guarded streams.

For limits of space, only the main proofs are sketched in this report.

2 Basic Notions: Substitutions and Unification

The notions of substitution and unification play a key role in the theory of logic programming. Here we summarize some definitions and results useful for our purposes, using as references [LMM87].

Let $\mathbf{A(V)}$ be an alphabet which contains function and constant symbols plus an arbitrary denumerable set of variables $\mathbf{V}$. From $\mathbf{A(V)}$ it is possible to define as usual the set of finite terms $\mathtt{Terms}(\mathbf{A(V)})$. The set of all ground terms is called the Herbrand Universe, $\mathtt{Herbrand}$. The set of variables occurring in any syntactic object o will be denoted $Vars(o)$.

The set of all substitutions $\mathtt{Subst}$ obtained from $\mathtt{Terms}\mathbf{A(V)}$ consist in all the mappings θ from $\mathbf{V}$ to $\mathtt{Terms}(\mathbf{A(\ V)})$ such that the domain of $\theta, \mathsf{dom}(\theta) = \{X \in \mathbf{V} : \theta X \not\equiv X\}$ is finite. The *application* $t\theta$ of θ to t is defined as the term obtained by replacing each variable X in t by $\theta(X)$. The set $\mathsf{ran}(\theta)$ is defined as $\bigcup_{X \in \mathsf{dom}(\theta)} Vars(\theta(X))$. The *composition* $\theta\theta'$ of θ and θ' is defined by $(\theta\theta')(X) \equiv (t\theta)\theta'$. The *restriction* of θ to a set of variable V is defined as $\theta_{\downarrow V} = \{X \leftarrow t : X \leftarrow t \in \theta \text{ and } X \in V\}$. θ is *ground* iff $\mathsf{ran}(\theta) = \emptyset$; *idempotent* iff $\theta\theta = \theta$, (equivalently, $\mathsf{dom}(\theta) \cap \mathsf{ran}(\theta) = \emptyset$). The set of idempotent substitutions is denoted $\mathtt{ISubst}$.

θ is said to be more general than $\theta', \theta \leq \theta'$ iff there exists σ such that $\theta\sigma = \theta'$. The associated equivalence relation is denoted $\approx$. Then $\leq$ is a partial order on the equivalence classes of $\mathtt{Subst}$, denoted $\mathtt{Subst}_{/\approx}$.

When substitutions are defined, a partial order on $\mathtt{Terms}(\mathbf{A(V)})$ can be defined as follows: if $t_1, t_2 \in \mathtt{Terms}(\mathbf{A(V)}), t_1 \leq t_2$ iff there is a substitution θ such that $t_1\theta \equiv t_2$.

An equation is an expression $\{s = t\}$, where s and t are terms. The class of equation sets from $\mathtt{Terms}(\mathbf{A}($

V)) is denoted Eqn. A *solution* of an equation $\{s = t\}$ is a ground substitution θ such that $s\theta \equiv t\theta$. The set of all solutions of $\{s = t\}$ is denoted $soln(\{s = t\})$. For an equation set $E, soln(E) = \cap_{\{s=t\}\in E} soln(\{s = t\})$.

An *unifier* for an equation is a substitution (possibly non-ground) μ such that $s\mu \equiv t\mu$. If an unifier exists for an equation E, E is said to be *unifiable*.The smallest unifier for an equation, its *mgu*, alway exists uniquely in $\mathrm{Eqn}_{/\approx}$.

E_1 is said to be more general than $E_2, E_1 \leq E_2$ iff $soln(E_2) \subseteq soln(E_1)$. The associated equivalence relation is denoted $\approx$. For equation sets, $\leq$ is a partial order on the equivalence classes of Eqn, denoted $\mathrm{Eqn}_{/\approx}$.

An equation set is in *solved form* (for short, s.f.equation set) when the lhs variables are all distinct and do not appear in the right side. The lhs variables of a s.f.equation set E, $elim(E)$, are called *eliminable*; the rhs variables and the variables which do not appear in $E, param(E)$, are called *parameters.* If the contrary is not explicitly stated, from now on it will be supposed that equation sets are always in solved form.

The following lemma reformulates results obtained in [LMM87] on the syntax of equivalent solved forms.

Lemma 1 *Let E_1 and E_2 be two s.f.equation sets such that $E_1 \approx E_2$. Then there exists a bijection ϕ between $elim(E_1)$ and $elim(E_2)$, such that, if $X \in elim(E_1) \cap elim(E_2), \phi(X) = X$. Moreover, for any $X \in elim(E_1) \cap param(E_2)$ there exists $Y \in elim(E_2) \cap param(E_1)$ such that $\{X = Y\} \in E_1$ and $\{Y = X\} \in E_2$.*

Lemma 1 can be generalized as follows.

Lemma 2 *Let E_1 and E_2 be two s.f.equation sets such that $E_1 \approx E_2$. Then,for any equation $\{X = t\} \in E_1$ we have:*

- *if t is a proper term, for a proper term t', there exists $\{X = t'\} \in E_2$;*
- *if t is a variable Y, one of the following is true:*
  - $\{X = Y\} \in E_2$;
  - $\{Y = X\} \in E_2$;
  - *for a variable $Z, \{Z = Y\} \in E_1$ and $\{X = Z, Y = Z\} \subseteq E_2$.*

3 C-units,c-unit sets and guarded streams

In this section, we study the concepts of c-units, c-unit sets and guarded streams, which we have introduced in Section.1.

First of all, we will give a formal definition of c-unit and we will present some operators and properties. A definition of meaning is also proposed. We will see that this technical framework can be extended to c-unit sets. However, the definition of meaning for c-unit sets will be quite more involved, because it is necessary to consider the semantic dependency between c-units, which induces a partial order over them. Later we will justify the introduction of guarded streams, as c-unit sets which verify some additional conditions of well-formedness.

Some minor difference exist between our definitions of c-units,c-unit sets and guarded streams and the original one given by [Mu90].

3.1 C-units

In the introduction, we have established as elementary computational unit (c-unit) a binding equation (or special symbol in $\{\Delta, \Phi\}$),prefixed by an equation set.

Def. 1 *A* **c-unit** *cu is a pair $< \bar{I}|O >$, such that:*

- *$\bar{I}$ (the input guard) is an equation set;*
- *O (the output part) is a single equation, such that $\bar{I} \cup O$ is consistent, or a special symbol in$\{\Delta, \Phi\}$.*

We define also the auxiliary functions $Ig(cu) = \bar{I}$ and $Ou(cu) = O$.

The special symbols $\{\Delta, \Phi\}$ give the *sign* of the c-unit. If, for a c-unit $cu, Ou(cu) \notin \{\Delta, \Phi\}$, the sign of cu is Ω.

C-units have a solved form, based on equations'solved form.

Def. 2 *A c-unit $cu =< \bar{I}|O >$ is in solved form when:*

- *$\bar{I}$ is in solved form;*
- *if $O \equiv \{Y = t\}$, there is no variable $X \in Vars(t) \cup \{Y\}$ such that $\{X = s\} \in \bar{I}$.*

The second condition is equivalent to say that the output part is "simplified" w.r.t. the input guard. A generic c-unit can be always reduced to a c-unit set in solved form. In the following, c-units are always assumed to be in solved form.

A s.f.equation set can be associated to a c-unit.

Def. 3 *For a c-unit cu, the substitution $\Sigma(cu)$ and the s.f.equation Eq(cu) associated to cu are defined as follows.*

$$Eq(cu) = \begin{cases} \bar{I}\{Y \leftarrow t\} \cup \{Y = t\} & if O \equiv \{Y = t\}; \\ \{\bar{I}\} & otherwise. \end{cases}$$

$$\Sigma(cu) = \{Y \leftarrow t : \{Y = t\} \in Eq(cu)\}$$

A first semantic characterization of a c-unit cu can be given by means of the solutions of $Eq(cu)$. We simply define $soln(cu) = soln(Eq(cu))$. If $soln(cu) \neq \emptyset$, then cu is *consistent.* However, it is also necessary to find an appropriate characterization for the input guard of a c-unit. We propose the following.

Def. 4 *A substitution μ is* **IG-admissible** *wrt a c-unit cu=$< \bar{I}|O >$ if there exists a solution σ for cu such that $\sigma = \sigma'\sigma''$, where σ' is an mgu for $\bar{I}, \sigma' \leq \mu \leq \sigma$.*

The set of IG-admissible substitutions for cu is denoted as adm(cu).

Solutions are used in the definition, because we want to exclude the possibility of having IG-admissible substitutions without corresponding solutions.

3.2 C-unit sets and guarded streams

When we pass to consider c-unit sets, we discover that the meaning is not easily defined. Before discussing the problem and proposing a solution, we give some technical definitions.

Def. 5 *If Cs is a c-unit set,*

- $\hat{Ig}(Cs) = \bigcup\{\bar{I} : \exists < \bar{I}|O > \in Cs\}, \hat{Ou}(Cs) = \bigcup\{O : \exists < \bar{I}|O > \in Cs, O \notin \{\Delta, \Phi\}\}$ *and* $\hat{Eq}(Cs) = \hat{Ig}(Cs) \cup \hat{Ou}(Cs)$ *(simple syntactic unions);*
- $Ig(Cs) = \mathsf{lub}(\hat{Ig}(Cs)), Ou(Cs) = \mathsf{lub}(\hat{Ou}(Cs)), Eq(Cs) = \mathsf{lub}(\hat{Eq}(Cs))$;
- $\Sigma(Ig(Cs)), \Sigma(Ou(Cs)), \Sigma(Cs)$ *are the substitutions corresponding to the lub's.*

Notice that $Ig(Cs), Ou(Cs), Eq(Cs)$ are unique only as equivalence classes in $(\mathrm{Eqn}_{/\approx}, \leq)$.

A c-unit set Cs to which no $\{\Delta, \Phi\}$ c-unit belongs has sign Ω. If a single $\{\Delta, \Phi\}$-signed c-unit belongs to Cs, Cs has the same sign of the c-unit. Otherwise Cs has sign *confused.*

The meaning of c-units was given by means of solutions and IG-admissible substitutions. Solutions for a c-unit set Cs are given by the extension $soln(Cs) = \bigcap_{cu \in Cs} soln(cu)$ (as before, a c-unit set is consistent iff $soln(Cs) \neq \emptyset$). For IG-admissibility, a different approach is needed, because if we try to derive the substitutions IG-admissible for a c-unit set just looking at its elements, we will notice some ambiguity.

In fact, to determine the IG-admissible substitutions of c-unit set Cs, it is not enough to consider Cs itself, but we should give a closer look to the sequences specified by Cs. These sequences are not simply permutations of the c-units in Cs, but should respect the partial order which arise naturally between c-units, w.r.t. their input guards.

Def. 6 *If Cs is a c-unit set and $cu, cu' \in Cs$, then:*

- $cu \sqsubseteq_1 cu'$ *iff* $Ig(cu)\Sigma(Ou(cu)) \leq Ig(cu')$;
- $cu \sqsubseteq cu'$ *iff* $cu \sqsubseteq_1 cu'$ *or* $cu \sqsubseteq_1 cu''$ *and* $cu'' \sqsubseteq cu'$;
- $cu \approx cu'$ *iff* $cu \sqsubseteq cu'$ *and* $cu' \sqsubseteq cu$.

It turns out that it is enough to use an intermediate object, less abstract then c-units sets, but more abstract then sequences.

Def. 7 *Given a c-unit set Cs, let a* **step** *s be a maximal non-empty subset of c-units such that, for all $cu, cu' \in s, Ig(cu) \approx Ig(cu')$. The step set of Cs is denoted Steps(Cs). Steps, being c-unit sets, inherit the definitions of sign and operators given for c-unit sets. The partial order $\sqsubseteq$ also is extended to steps. If $s, s' \in Steps(Cs), s \sqsubseteq_1 s'$ iff $Ig(s)\Sigma(Ou(s)) \leq Ig(s')$. Also:*

- $s \sqsubseteq s'$ *iff* $s \sqsubseteq_1 s'$ *or* $s \sqsubseteq_1 s''$ *and* $s'' \sqsubseteq s'$;
- $s \approx s'$ *iff* $s \sqsubseteq s'$ *and* $s' \sqsubseteq s$.

Then, instead of a a sequence (a total ordering of c-units), we consider a total ordering of steps.

Def. 8 *Given a c-unit set Cs, A* **simulation** $\mathcal{S} = [s_1, ..., s_m]$ *is a sequence of steps of Cs such that:*

- *for* $1 \leq i < j \leq m, s_j \not\sqsubseteq s_i$;
- *for all* $s \in Steps(Cs)$ *such that, for* $s' \in \mathcal{S}, s \sqsubseteq s', s \in \mathcal{S}$;
- *if* $s \in \mathcal{S}$ *has sign* Δ, Φ *or confused, no other* $s' \in \mathcal{S}$ *has sign* Δ, Φ *or confused, and* $s \equiv s_m$.

If Cs has sign Ω, a simulation is maximal iff $m = |Steps(Cs)|$. Otherwise, it is maximal iff s_m has sign Δ, Φ or confused. The set of maximal simulations of Cs is denoted as Sim(Cs).

From now on, by simulation we will always intend a maximal simulation. At this point, IG-admissibility is defined as follows.

Def. 9 *A substitution μ is* **IG-admissible** *for a c-unit set Cs iff there is a simulation $\mathcal{S} = [s_1, ..., s_m]$ of Cs such that:*

- $\mu_1 = \mu$ *is IG-admissible for all* $cu \in s_1$;
- *for* $2 \leq i \leq m, \mu_i = \mathsf{lub}(\mu_{i-1}, \Sigma(s_{i-1}))$ *is IG-admissible for all* $cu \in s_i$.

The set of IG-admissible substitutions for a c-unit set Cs, through $\mathcal{S}$ is denoted adm(Gs,$\mathcal{S}$).

Guarded streams will raise naturally as c-unit sets over which some syntactic conditions guarantee that the information carried by the occurrences of the same variable in different c-units is monotonic w.r.t. the $\sqsubseteq$-ordering of c-units.

Def. 10 *A consistent c-unit set Gs is a* **guarded stream** *iff is satisfies the following conditions on $cu \in Gs$:*

1. *if, for $cu' \in Gs, Ig(cu) \approx Ig(cu')$, then $Ig(cu') \equiv Ig(cu)$;*
2. *for any variable $X \in Vars(Gs)$:*
   (a) *if there exists $cu' \in Gs$ such that, for $t, t' \in$ $\texttt{Terms}, \{X = t\} \in Ig(cu), \{X = t'\} \in Ig(cu')$, then:*
      - *if $t = t', cu \sqsubset cu', cu = cu'$ or $cu' \sqsubset cu$;*
      - *if $t \neq t'$, $t < t'$, and $cu \sqsubset cu'$ or $t' < t$, and $cu' \sqsubset cu$;*

   (b) *if $cu =< \bar{I}|X = t >$, there is no $cu' \in Gs, \{X = t'\} \in cu'$;*

   (c) *if $Y = t[X] \in Ig(cu)$ or $Ou(cu)$, for $cu' \in Gs$:*
      - *if $\{X = t'\} \in Ig(cu')$, then $cu \sqsubset cu'$;*
      - *if $\{X = t'\} \in Ou(cu')$, then $cu \sqsubseteq cu'$;*
3. *there is at most a c-unit $cu \in Gs$ such that $Ou(cu) \in \{\Delta, \Phi\}$, and for cu there is no $cu' \in Gs$ such that $cu' \sqsubset cu$.*

As a guarded stream Gs is a c-unit set, sign, $Eq(Gs), \hat{E}q(Gs), \Sigma(Gs)$ and the like are defined.

The three conditions used in Def. 10 can be understood as follows.

When equivalent input guards belong to c-units in the same guarded stream, Condition.1 forces them to be syntactically identical. This allows the correct functioning of Condition.2 on the occurrences of variables in the guarded stream.

The effects of Condition.2. are summarized by the following lemma, which expresses the key property of guarded streams.

Lemma 3 *Given a guarded stream Gs and $X \in Vars(Gs)$ let $StEl(X, Gs) \subseteq Steps(Gs)$ be the step set where X appears as an eliminable variable and $StPa(X, Gs) \subseteq Steps(Gs)$ be the step set where X appears as a parameter. Then $StEl(X, Gs)$ is totally ordered and ,for $s \in StPa(X, Gs)$ and $s' \in StEl(X, Gs), s \sqsubseteq s'$.*

Finally, Condition.3 guarantees that no guarded stream has sign confused. Intuitively, this corresponds to say that the termination mode of a guarded stream is uniquely defined.

4 Equivalence Relations

As we have seen in Section.1, solutions induce a partial order on equation sets. We would like to introduce a similar semantic equivalence on c-units and guarded streams.

Clearly, such equivalence should be based on the comparisons of the sign, solutions and IG-admissible substitutions of the c-units and guarded streams.

The naive approach, which makes two c-units or guarded stream equivalent if they have same sign, solutions and IG-admissible substitutions work for c-units, but it is unsatisfactory for guarded streams. We need in fact a finer equivalence, which respects the granularity of the input guards in the guarded streams' c-units. We will see in Section 4.1 that a step-wise equivalence, based on the simulations, will satisfy this criterium.

[LMM87] gives also a syntactic characterization of equivalent equation sets.

Lemma 4 *(Theorem 4,[LMM87]) Let E_1 and E_2 be s.f.equation sets. Then $E_1 \approx E_2$ iff there is a subset $\{X_1 = Y_1, ..., X_k = Y_k\}$ of E_1, where the Y_i's are distinct variables, such that $E_2 \equiv E_1\{X_1 \leftarrow Y_1, ..., X_k \leftarrow Y_k, ...Y_1 \leftarrow X_1, ..., Y_k \leftarrow X_k\}$.*

A similar characterization can be found for c-units. Two c-units cu and cu' will be equivalent iff they have the same sign and there is a substitution θ which verify Lemma 4 between $Ig(cu)$ and $Ig(cu')$ and a substitution θ' which verify Lemma 4 between $Ou(cu)$ and $Ou(cu')\theta$.

In the case of guarded streams, this two-stage syntactic equivalence becomes fairly more complicated. In fact, when we want to check the syntactic equivalence between two guarded streams Gs and Gt, we should find a bijection ϕ between their c-units such that, $cu \in Gs$ is syntactically equivalent to the corresponding $\phi(cu) \in Gt$. The syntactic representation of the input guards and output parts of cu and $\phi(cu)$ depend also from the c-units which $\sqsubset$-precede them, respectively in Gs and Gt. It is straightforward to notice that these c-units should be pairwise syntactically equivalent and that the syntactic equivalence of cu and $\phi(cu)$ depends on the way variables are equated in these c-units. To keep track of these equated variables by means of substitutions is quite troublesome. However, a different approach is possible. Let us go back to to Lemma 4 and look at it from a different perspective. Let us suppose that we build a partition of the variables in E_1 such that two variables belong to the same set in the partition iff they are directly or indirectly equated in E_1. The partition is clearly well defined; we call its sets $R - sets$. In any solution for E_1, all the variables in the same R-set will be assigned to the same ground term; these variables are "interchangeable". Then Lemma 4 says that to any occurrence of a variable in E_2 it corresponds an occurrence of a variable in E_1 which belong to the same R-set.

Then syntactic equivalence of guarded streams can be defined using R-sets instead of substitutions. All we need is to build the R-sets of the current c-unit from the R-sets of preceding c-units and check that all the occurrences of variables in the two c-units correspond and belong to the same R-set.

The syntactic equivalence for c-units and guarded streams based on R-sets is presented in Section 4.2. We will prove that, as it should be expected, semantic and syntactic equivalence correspond in all situations of interest.

4.1 Semantic equivalence

It is straightforward to define a semantic partial order for c-units.

Def. 11 *Given two c-units cu and cu' with the same sign, cu$\preceq$cu' iff soln(cu) $\subseteq$ soln(cu') and adm(cu) $\subseteq$ adm(cu'). Then cu $\approx$ cu' iff cu$\preceq$cu' and cu'$\preceq$cu.*

We have discussed before that a similar equivalence, based only on solutions and IG-admissible substitutions would be too coarse for guarded streams, and we need to consider the simulations.

Def. 12 *Given two guarded streams Gs and Gt, if $\mathcal{S} = [s_1, ..., s_m] \in Sim(Gs)$ and $\mathcal{T} = [t_1, ..., t_m] \in Sim(Gt)(\mathcal{S} \approx \mathcal{T})$ iff they have the same sign and, for $1 \leq i \leq n = m, soln(\bigcup_{j=1}^{i} s_j) = soln(\bigcup_{j=1}^{i} t_j)$, and $adm(\bigcup_{j=1}^{i} s_j) = adm(\bigcup_{j=1}^{i} t_j)$.*

The preorder on guarded streams is defined as follows.

Def. 13 *If Gs and Gt are two guarded streams, $Sim(Gs) \preceq Sim(Gt)$ iff for all $\mathcal{S} \in Sim(Gs)$, there exists $\mathcal{T} \in Sim(Gt)$ such that $\mathcal{S} \approx \mathcal{T}$; $Sim(Gs) \approx Sim(Gt)$ iff $Sim(Gs) \preceq Sim(Gt)$ and $Sim(Gt) \preceq Sim(Gs)$.*

$Gs \preceq Gt$ iff $Sim(Gs) \preceq Sim(Gt); Gs \approx Gt$ iff $Sim(Gs) \approx Sim(Gt)$.

A final extension is to classes of guarded streams.

Def. 14 *Given two classes of guarded streams Gs and GT, $GS \preceq GT$ iff for all $Gs \in GS$, there exists $Gt \in GT$ such that $Gs \approx Gt; GS \approx GT$ iff $GS \preceq GT$ and $GT \preceq GS$.*

The overloading of "$\preceq$" and "$\approx$" stresses the relation between respectively the partial orders and the equivalence relations which were defined, and should not cause ambiguity.

4.2 Syntactic equivalence

The syntactic equivalence based on R-sets(R-equivalence) is first defined for equation sets and then extended to c-units, guarded streams and sets of guarded streams. Each of these stages correspond to an increase in syntactic complexity. Then the R-equivalence will also be verified by means of more complex structures based on the R-sets.

4.2.1 R-equivalence for equation sets

The first step toward an equivalence based on R-sets between equation sets is to formalize the concept of R-set itself. We remind here that this equivalence will be later used for input guards and output parts of c-units, for which the R-sets will depend also on the R-sets of the $\sqsubseteq$-smaller/equal c-units.

Then an equation set will only impose some restrictive conditions on its possible R-sets.

Def. 15 *Given an equation set E in solved form, a class $\mathcal{R}$ of variable sets (**R-sets**) is an **R-class for** E , iff for any variable $X \in Vars(E)$, there exists uniquely an R-set $R(X) \in \mathcal{R}$ which verifies the following conditions:*

- $X \in R(X)$;
- *if* $\{X = t\} \in E$ *and* $t \equiv Y \in Vars, R(X) = \mathcal{R}(Y)$;
- *if* $\{Y = X\} \in E, R(X) = R(Y)$.

Notice that it is possible to have an R-set $R \in \mathcal{R}$ such that for no $X \in Vars(E), R = R(X)$. Also, it is possible to have $\mathcal{R}(X) \subseteq R(Y)$, even if there is no equation $\{X = Y\}$.

The R-equivalence on terms,equations and equation sets can be defined now as follows. Special symbols Δ and Φ are considered for this purpose as reserved constant symbols.

Def. 16 R-equivalence *Given two terms t and t' , $t \stackrel{\mathcal{R}}{\sim} t'$ iff:*

- *t and t' are the same constant symbol;*
- $t \equiv X, t' \equiv Y, \{X,Y\} \subset Vars$ *and for* $R \in \mathcal{R}, X, Y \in R$;
- $t \equiv f(t_1, ..., t_m), t' \equiv f(t'_1, ..., t'_m)$, *and for* $1 \leq i \leq m, t_i \stackrel{\mathcal{R}}{\sim} t'_i$.

Given two equations E and E',if $\mathcal{R}$ is an R-class for E and E', $E \stackrel{\mathcal{R}}{\sim} E'$ iff $lhs(E) \stackrel{\mathcal{R}}{\sim} lhs(E')$ and $rhs(E) \stackrel{\mathcal{R}}{\sim} rhs(E')$.

Given two equation sets E and E', $E \stackrel{\mathcal{R}}{\subseteq} E'$ iff, for any equation $\{X = t\} \in E$ there is an equation $\{Y =$

$s\} \in E'$ *such that* $\{X = t\} \stackrel{\mathcal{R}}{\sim} \{Y = s\}$. *If* $E \stackrel{\mathcal{R}}{\subseteq} E'$ *and* $E' \stackrel{\mathcal{R}}{\subseteq} E$, *then* $E \stackrel{\mathcal{R}}{\sim} E'$.

If a partial order is defined on R-classes, it is possible to associate uniquely to an equation set E its minimal R-class. Before we define the partial order and we show the existence of the lub of two R-classes.

Lemma 5 *Given two R-classes* $\mathcal{R}_1$ *and* $\mathcal{R}_2, \mathcal{R}_1 \subseteq \mathcal{R}_2$ *iff, for all* $R \in \mathcal{R}_1$, *there exists* $R' \in \mathcal{R}_2, R \subseteq R'$. *Then* $\mathcal{R} = \mathrm{lub}(\mathcal{R}_1, \mathcal{R}_2)$ *is the minimal R-class which satisfies the following conditions:*

- *for any* $R \in \mathcal{R}_k, k \in \{1,2\}$, *there exists* $R' \in \mathcal{R}, R \subseteq R'$;
- *for any* $R \in \mathcal{R}, R = \bigcup_{i=1}^{n} R_i, n \leq |\mathcal{R}_1| + |\mathcal{R}_2|, R_i \in \mathcal{R}_{k_i}$, *where* $k_i \in \{1,2\}$;
- *for any pair* $R_1 \in \mathcal{R}_1$ *and* $R_2 \in \mathcal{R}_2$ *for which* $R_1 \cap R_2 \neq \emptyset$, *there exists* $R \in \mathcal{R}$ *such that* $R_1, R_2 \subseteq \mathcal{R}$.

The lub of a set of R-classes $\{\mathcal{R}_i : 1 \leq i \leq n\}$ is denoted $\mathrm{lub}(\{\mathcal{R}_i : 1 \leq i \leq n\})$.

Lemma 6 *Given an s.f.equation set E, let* $\mathcal{R}_E$ *be the R-class of E such that, for any* $R \in \mathcal{R}_E, \{X,Y\} \subseteq R$ *iff* $\{X = Y\} \in E$ *or* $\{Y = X\} \in E$ *or, for a variable* $Z, \{X = Z, Y = Z\} \in E$. $\mathcal{R}_E$ *is well-defined and is the minimal R-class for E.*

We define also minimal R-classes for the special symbols Δ and Φ : $\mathcal{R}_\Delta = \mathcal{R}_\Phi = \emptyset$. A simple result which connects the partial order on R-classes to the equivalence is given by the following lemma.

Lemma 7 *Given two equation sets* E_1 *and* E_2, *if* $\mathcal{R} \subseteq \mathcal{R}'$ *and* $E_1 \stackrel{\mathcal{R}}{\sim} E_2$, *then* $E_1 \stackrel{\mathcal{R}'}{\sim} E_2$.

It is possible also to give a nice characterization of the relation between R-equivalence and semantic equivalence.

Proposition 1 *Given two equation sets* E_1 *and* E_2, $E_1 \approx E_2$ *iff* $\mathcal{R}_{E_1} = \mathcal{R}_{E_2} = \mathcal{R}$ *and* $E_1 \stackrel{\mathcal{R}}{\sim} E_2$.

Corollary 1 *If* $E_1 \approx E_2$, *then for any R-class* $\mathcal{R}$ *for* E_1 *and* E_2, $E_1 \stackrel{\mathcal{R}}{\sim} E_2$.

It is evident that the inverse is not true: for some R-class R-class $\mathcal{R}$, it can be $E_1 \stackrel{\mathcal{R}}{\sim} E_2$ and $E_1 \not\approx E_2$.

4.2.2 R-equivalence for c-units

R-equivalence can be defined quite directly to c-units in terms of the R-equivalence of the equation sets and special symbols which compose them.

Def. 17 *A pair of classes of variable sets* $\tilde{\mathcal{R}} = (\mathcal{R}_1, \mathcal{R}_2)$ *is an* **R-pair for a c-unit** *cu iff:*

- $\mathcal{R}_1 \subseteq \mathcal{R}_2$;
- $\mathcal{R}_1$ *is an R-class for Ig(cu);*
- $\mathcal{R}_2$ *is an R-class for Ou(cu).*

We define also $\mathrm{proj}_1(\tilde{\mathcal{R}}) = \mathcal{R}_1$ *and* $\mathrm{proj}_2(\tilde{\mathcal{R}}) = \mathcal{R}_2$.

Given two c-units cu and cu', cu $\stackrel{\tilde{\mathcal{R}}}{\sim}$ *cu' iff Ig(cu)* $\stackrel{\mathcal{R}_1}{\sim}$ *Ig(cu') and Ou(cu)* $\stackrel{\mathcal{R}_2}{\sim}$ *Ou(cu').*

The first condition takes in account the fact that the representation of the output part depends on the representation of the input guard.

For c-units also, a uniquely defined minimal R-pair exists.

Lemma 8 *If* $\tilde{\mathcal{R}}$ *and* $\tilde{\mathcal{R}}'$ *are two R-pairs,* $\tilde{\mathcal{R}} \subseteq \tilde{\mathcal{R}}'$ *iff* $\mathrm{proj}_1(\tilde{\mathcal{R}}) \subseteq \mathrm{proj}_1(\tilde{\mathcal{R}}')$ *and* $\mathrm{proj}_2(\tilde{\mathcal{R}}) \subseteq \mathrm{proj}_2(\tilde{\mathcal{R}}')$.

Given a c-unit cu, let $\tilde{\mathcal{R}}_{cu}$ *be the R-pair of cu such that* $\mathrm{proj}_1(\tilde{\mathcal{R}}) = \mathcal{R}_{Ig(cu)}$ *and* $\mathrm{proj}_2(\tilde{\mathcal{R}}) = \mathrm{lub}(\mathcal{R}_{Ig(cu)}, \mathcal{R}_{Ou(cu)})$. *Then* $\tilde{\mathcal{R}}_{cu}$ *is well-defined and is the minimal R-pair of cu.*

4.2.3 R-equivalence for guarded streams

The R-equivalence between guarded streams is verified showing that the c-units in the guarded streams are pairwise equivalent. We remind that, if $cu \sqsubseteq cu'$ and $cu' \sqsubseteq cu, cu \doteq cu'$.

Def. 18 *A class* $\Re$ *of R-pairs is a* **R/p-class for a guarded stream** *Gs iff for any* $cu \in Gs$, *there exists uniquely an R-pair* $\tilde{\mathcal{R}}(cu)$ *such that:*

- $\mathrm{proj}_1(\Re(cu))$ *is an R-class for Ig(cu);*
- $\mathrm{proj}_2(\Re(cu))$ *is an R-class for* $\{Ou(cu') : cu \doteq cu'\}$;
- *for any* $cu' \in Gs$ *such that* $cu' \sqsubset cu, \mathrm{proj}_2(\Re(cu')) \subseteq \mathrm{proj}_1(\Re(cu))$.

Given two guarded streams Gs and Gt, if $\Re$ *is an R/p-class for Gs and Gt,* $Gs \stackrel{\Re}{\subseteq} Gt$ *iff, for any* $cu \in Gs$, *there exists* $cu' \in Gt, cu \stackrel{\Re(cu)}{\sim} cu'$. $Gs \stackrel{\Re}{\sim} Gt$ *iff* $Gs \stackrel{\Re}{\subseteq} Gt$ *and* $Gt \stackrel{\Re}{\subseteq} Gs$.

Once more, we prove the existence of the minimal R/p-class for a guarded stream.

Lemma 9 *Given two R/p-classes $\Re$ and $\Re'$, $\Re \subseteq \Re'$ iff, for any $\tilde{\mathcal{R}} \in \Re$, there exists $\tilde{\mathcal{R}}' \in \Re', \tilde{\mathcal{R}} \subseteq \tilde{\mathcal{R}}'$.*

Given a guarded stream Gs, let $\Re_{Gs}$ be the R/p-class of Gs such that for any $cu \in Gs$, we have :

- $\mathsf{proj}_1(\Re_{Gs}(cu)) = \mathsf{lub}(\mathsf{lub}(\{\mathsf{proj}_2(\Re_{Gs}(cu')) : cu' \sqsubset cu\}), \mathcal{R}_{Ig(cu)})$;
- $\mathsf{proj}_2(\Re_{Gs}(cu)) = \mathsf{lub}(\mathsf{proj}_1(\Re_{Gs}(cu)), \mathsf{lub}(\{\mathcal{R}_{Ou(cu')} : cu \doteq cu'\}))$, *and for any $\tilde{\mathcal{R}} \in \Re_{Gs}$ there is a $cu \in Gs, \tilde{\mathcal{R}} = \Re_{Gs}(cu)$.*

Then $\Re_{Gs}$ is well-defined and is the minimal R/p-class for Gs.

In practice, we will always be interested to verify R-equivalence w.r.t. R/p-classes which are minimal for at least one of the guarded streams involved.

The following lemma introduces some additional conditions on the R/p-class which verifies R-equivalence between two guarded streams, and show that, for this R/p-class, R-equivalence can be extended to the steps in the guarded streams. We will use this result later to prove that, when the additional conditions hold, the two guarded streams are also semantically equivalent.

Lemma 10 *Let Gs and Gt be two guarded streams such that there exists a bijection $\phi : Gs \mapsto Gt$, for which, if $cu \in Gs, \Re_{Gs}(cu) = \Re_{Gt}(\phi(cu)) = \tilde{\mathcal{R}}$ and $cu \stackrel{\tilde{\mathcal{R}}}{\sim} \phi(cu)$. Then:*

1. *$\Re_{Gs} = \Re_{Gt} = \Re$ and $Gs \stackrel{\Re}{\sim} Gt$;*
2. *there exists a bijection ψ : $Steps(Gs) \mapsto Steps(Gt)$, for which, if $s \in Steps(Gs)$, there exists $\psi(s)$ such that, if $\Re_{Gs}(s) = \{\Re_{Gs}(cu) : cu \in s\}, s \stackrel{\Re_{Gs}(s)}{\sim} \psi(s)$.*

Finally, we can give the main result of the section.

Proposition 2 *Let Gs and Gt be two guarded streams such that there exists a bijection $\phi : Gs \mapsto Gt$, for which, if $cu \in Gs, \Re_{Gs}(cu) = \Re_{Gt}(\phi(cu)) = \tilde{\mathcal{R}}$ and $cu \stackrel{\tilde{\mathcal{R}}}{\sim} \phi(cu)$. Then $Gs \approx Gt$.*

It is possible a straightforward extension to sets of guarded streams.

Def. 19 *A collection $\mathbf{R}$ of R/p-classes is an R-collection for a set of guarded streams GS iff, for any $Gs \in GS$, there exists uniquely an R/p-class $\mathbf{R}(Gs)$ which is an R/p-class for Gs. If GT is another set of guarded streams for which $\mathbf{R}$ is an R-collection, $GS \stackrel{\mathcal{R}}{\subseteq} \mathbf{R} \stackrel{\mathcal{R}}{\subseteq} GT$ iff, for any $Gs \in GS$, there exists $Gt \in GT$ such that $Gs \stackrel{\mathbf{R}(Gs)}{\sim} Gt$. $GS \stackrel{\mathbf{R}}{\approx} GT$ iff $Gs \stackrel{\mathbf{R}(Gs)}{\sim} Gt$ and $Gt \stackrel{\mathbf{R}(Gs)}{\sim} Gs$.*

A R-collection $\mathbf{R}_{GS}$ for a set of guarded streams GS is minimal iff for any $\Re \in \mathbf{R}_{GS}$, there exists $Gs \in GS$ such that $\Re = \mathbf{R}_{GS}(Gs)$ and for all $Gs \in GS$,$\mathbf{R}_{GS}(Gs)$ is minimal.

Proposition 2 leads to the following result.

Proposition 3 *Given two sets of guarded streams GS and GT,if $GS \stackrel{\mathbf{R}_{GS}}{\approx} GT$, then $GS \approx GT$.*

5 Conclusion

In the previous sections we have introduced definitions and results about semantic and syntactic equivalences of guarded streams.

It is possible to proof that both equivalences are decidable on finite guarded streams. Also, the semantic equivalence on guarded streams can be used to discuss the semantic equivalence of programs and it allows to measure the well-definedness of operators defined on guarded streams.

We are getting close to full-abstractness for the success semantics proposed in [Mu90]. For the deadlock and failure semantics we would like to adopt as target program a completed CLP-program, analogous to the Clark-completed program for sequential Prolog.

We are also considering the extension of the equivalence relations to a domain, similar to guarded streams, which can denote perpetual computations.

Acknowledgement The authors thank Prof.T.Sakabe of Nagoya University for useful comments on earlier drafts.

References

[BoPa90a] F. S.de Boer, C. Palamidessi. *Concurrent Logic Programming: Asynchronism and Language Comparison*, in: (S.Debray,M.Hermenegildo eds.) Logic Programming, pp.175-194, Proc.s 1990 North-American Conference

[BoPa90b] F. S.de Boer, C. Palamidessi. *On the Asynchronous Nature of Communication in Concurrent Logic Languages: A Fully Abstract Model based on Sequences*, in:(J.C.M.Baeten,J.W.Klop eds.) CONCUR '90, LNCS 458, pp.99-114, Springer-Verlag 1990

[GaLe92] M.Gabbrielli, G.Levi. *Unfolding and Fixpoint Semantics of Concurrent Constraint Logic Programs.* T.C.S. 105,pp.85-128, Elsevier 1992

[GMS89] H.Gaifman,M.J.Maher,E.Shapiro. *Reactive Behavior Semantics for Concurrent Constraint Logic Programs.*In: (E.Lusk,R.Overbeek eds.) North American Conf. on Logic Programming,pp.535-572,1989

[GKLS88] R.Gerth, M.Codish, Y.Lichtenstein, E.Shapiro. *Fully Abstract Denotational Semantics for Flat Concurrent Prolog* Proc. Third IEEE Symp. on Logic in Computer Science, pp.320-335, IEEE Computer Society Press 1988

[KaMu93] T. Kato, M. Murakami. *An Or-Compositional Semantics of Guarded Horn Clauses.* In: Joint Symposium on Parallel Processing 1993, 1993

[LMM87] J.-L.Lassez, M.J.Maher, K.Marriott. *Unification Revisited* in:(J.Minker ed.) Foundations of Deductive Databases and Logic Programming,pp.587-625, Ed.Kaufmann, Los Altos CA. 1987

[Le88b] G.Levi. *Models, Unfolding Rules and Fixpoint Semantics.* (R.A.Kowalski, K.A. Bowen eds.) Proc. Fifth Int'l Conference on Logic Programming, pp.1649-1665, The MIT Press, Cambridge, Ma. 1988.

[Ll87] J.W.Lloyd. *Foundations of Logic Programming.Second, extended edition* Springer V., 1987

[Mu90] M.Murakami. *A Declarative Semantics of Flat Guarded Horn Clauses for Programs with Perpetual Processes.* T.C.S. 75,pp.67-83, Elsevier 1990

[SPR91] V.A.Saraswat, M.Rinard,P.Panangaden. *Semantic foundations of concurrent constraint programming.* Proc.18th ACM Symposium on Principles of Programming Languages, pp.333-352, ACM 1991

[Ue88] K.Ueda. *Guarded Horn Clauses: A Parallel Logic Programming Language with the Concept of a Guard.*In:(M.Nivat,K.Fuchi eds.) Programming of Future Generation Computers, pp.441-456, North-Holland 1988

Appendix:Proofs

Proof Proposition 1 We have to prove:

$\Rightarrow$ if $E_1 \approx E_2$, then $\mathcal{R}_{E_1} = \mathcal{R}_{E_2} = \mathcal{R}$ and $E_1 \overset{\mathcal{R}}{\sim} E_2$;

$\Leftarrow$ if $\mathcal{R}_{E_1} = \mathcal{R}_{E_2} = \mathcal{R}$ and $E_1 \overset{\mathcal{R}}{\sim} E_2$, then $E_1 \approx E_2$.

Proof.$\Rightarrow$ First of all, let us prove $\mathcal{R}_{E_1} = \mathcal{R}_{E_2}$ by absurd, i.e. supposing $\mathcal{R}_{E_1} \neq \mathcal{R}_{E_2}$.

It is enough to examine one of two symmetric possibilities:

- for $R \in \mathcal{R}_{E_1}(\mathcal{R}_{E_2})$, there exists $X \in R, X \notin \bigcup\{R : R \in \mathcal{R}_{E_2}\}(\bigcup\{R : R \in \mathcal{R}_{E_1}\})$ or $X \in R', R' \in \mathcal{R}_{E_2}(\mathcal{R}_{E_1}), R' \neq R$.

If $X \notin \bigcup\{R : R \in \mathcal{R}_{E_2}\}$, then the solutions of E_1 and E_2 are clearly different, against hypothesis of equivalence.

If $X \in R'R' \in \mathcal{R}_{E_2}, R' \neq R$, then there is a variable $Y \in R' - R$ or $Y \in R - R'$. We treat only the first case for symmetry.

In E_2, as X and Y belong to the same R-set, for any possible ground term t, there is a solution $\sigma \in soln(E_2)$, such that $\{X \leftarrow t, Y \leftarrow t\} \subseteq \sigma$. However, in E_1, they belong to different R-sets. If one of these sets is a singleton, one of the variables can assume less values and E_1 does not have all the solutions of E_2. Otherwise, it is possible to assign to X and Y distinct values. In both cases, $E_1 \not\approx E_2$.

Now let us prove $E_1 \overset{\mathcal{R}}{\sim} E_2$. By symmetry it is enough to prove $E_1 \overset{\mathcal{R}}{\subseteq} E_2$.

Let us consider an equation $\{X = t\} \in E_1$. We will prove the existence of an equation $\{Y = s\} \in E_2$ such that $\{X = t\} \overset{\mathcal{R}}{\sim} \{Y = s\}$.

By Lemma 2, as $E_1 \approx E_2$, one of the following statements is true:

[a] t is a proper term and $\{X = t'\} \in E_2$;

[b] t is a variable W, and then one of the following is true:

- $\{X = W\} \in E_2$;
- $\{W = X\} \in E_2$;
- $\{X = Z, W = Z\} \in E_2$.

Case [b] is quite simple, because for the three alternatives we have respectively $\{X = W\} \overset{\mathcal{R}}{\sim} \{X = W\}, \{X = W\} \overset{\mathcal{R}}{\sim} \{W = X\}, \{X = W\} \overset{\mathcal{R}}{\sim} \{X = Z\}$ and $\{X = W\} \overset{\mathcal{R}}{\sim} \{W = Z\}$.

For Case [a], clearly $X \overset{\mathcal{R}}{\sim} X$ and it is enough to prove $t \overset{\mathcal{R}}{\sim} t'$. As for any solution $t\sigma \equiv X\sigma$ and $t'\sigma \equiv X\sigma$, it should be $t\sigma \equiv t'\sigma$. Notice that this condition holds also on the corresponding subterms of t and t'.

We prove, by induction on the depth of t and t', that this condition implies $t \overset{\mathcal{R}}{\sim} t'$. For $\mathsf{depth}(t) = 1, t$ can only be a constant,and clearly t' should be the same constant, otherwise $E_1 \not\approx E_2$. If $\mathsf{depth}(t) = 2, t = f(t_1, ..., t_k)$ where $t_i, 1 \leq i \leq k$ can be a constant a or a variable V_i. Clearly it must be $t' = f(t'_1, ..., t'_k)$.

If t_i is a constant a, $t'_i = a$. Otherwise, if $t_i \equiv V_i, t'_i = W_i$, such that in any solution V_i and W_i have the same value; otherwise $E_1 \not\approx E_2$.

This is possible only if $\{W_i = V_i\} \in E_1$ and, as a consequence (Lemma 1), $\{V_i = W_i\} \in E_2$. Then $W_i \overset{\mathcal{R}}{\sim} V_i$. Then for all $i, 1 \leq i \leq k, t_i \overset{\mathcal{R}}{\sim} t'_i$ and $f(t_1, ..., t_k) \overset{\mathcal{R}}{\sim} f(t'_1, ..., t'_k)$. The induction step on $\mathsf{depth}(t) = m+1$ is straightforward, because t and t' must have the same outermost symbol and the same arity, and to prove $t_i \overset{\mathcal{R}}{\sim} t'_i$ we can use the inductive hypothesis.

Proof.$\Leftarrow$ By symmetry, it is enough to prove $soln(E_1) \subseteq soln(E_2)$. Let it be $\sigma \in soln(E_1)$; we will prove $\sigma \in soln(E_2)$. $\sigma \in soln(E_2)$ iff it is a solution for any equation in E_2. Let it be $\{X = t\} \in E_2$. As $E_1 \overset{\mathcal{R}}{\sim} E_2$, there exists $\{Y = s\} \in E_1$ such that $\{X = t\} \overset{\mathcal{R}}{\sim} \{Y = s\}$, $X \overset{\mathcal{R}}{\sim} Y$ and $t \overset{\mathcal{R}}{\sim} s$. Then we will prove $X\sigma \equiv Y\sigma$ and $t\sigma \equiv s\sigma$. Then, as $X\sigma \equiv t\sigma, Y\sigma \equiv s\sigma$; σ is a solution for $\{X = t\}$. This ends the proof.

If $X \overset{\mathcal{R}}{\sim} Y$, then $\{X,Y\} \in R, R \in \mathcal{R}$. Then in E_1 it must be $X \equiv Y$ (and of course $X\sigma = Y\sigma$) or, for Lemma 6 we should have:

- $\{X = Y\}$ or $\{Y = X\} \in E_1$ and $X\sigma = Y\sigma$;
- $\{X = Z, Y = Z\} \in E_1$ and $X\sigma = Y\sigma = Z\sigma$.

To prove $t\sigma \equiv s\sigma$, knowing $t \overset{\mathcal{R}}{\sim} s$, we use again an induction on the depth of t and s. Notice that it should be $\mathsf{depth}(t) = \mathsf{depth}(s)$. If $\mathsf{depth}(t) = 1$, and t is a constant, s is the same constant and $t\sigma \equiv s\sigma$. If $t \equiv V$, $s \equiv W$; by the same observations given on $\{X,Y\}, V\sigma \equiv W\sigma$.

For $\mathsf{depth}(t) = k > 1, t \equiv f(t_1,...,t_k), s \equiv f(s_1,...,s_{k'})$ and $t \overset{\mathcal{R}}{\sim} s$ iff $f \equiv f', k \equiv k'$ and, for $q \leq i \leq k, t_i \overset{\mathcal{R}}{\sim} s_i$. By inductive hypothesis, $t_i\sigma \equiv s_i\sigma$ and $t\sigma \equiv s\sigma$. □

Proof (Sketch) Lemma 10 The first part of the lemma is trivial. For any $\tilde{\mathcal{R}} \in \Re_{Gs}$, there exists $cu \in Gs$ such that $\tilde{\mathcal{R}} = \Re_{Gs}(cu) = \Re_{Gt}(\phi(cu))$. Then $\Re_{Gs} \subseteq \Re_{Gt}$. As ϕ is a bijection, the inverse also holds and $\Re_{Gs} = \Re_{Gt}$. For any $cu \in Gs, cu \overset{\tilde{\mathcal{R}}_{Gs}(cu)}{\sim} \phi(cu)$ and $Gs \overset{\Re}{\subseteq} Gt$; also, using ϕ^{-1}, $Gt \overset{\Re}{\subseteq} Gs$ and $Gs \overset{\Re}{\sim} Gt$.

We are left with the second part of the Lemma.

Let it be $cs_1, cs_2 \in s, s \in Steps(Gs)$ and $ct_1, ct_2 \in Gt$, such that $cs_1 \overset{\Re_{Gs}(cs_1)}{\sim} ct_1$ and $cs_2 \overset{\Re_{Gs}(cs_2)}{\sim} ct_2$. Then we will prove that, for any equation in $Ig(ct_1)$, the same equation is in $Ig(ct_2)$ and, as the reverse can be proved in the same way, we clearly have $Ig(ct_1) \equiv Ig(ct_2)$. This will prove that the image of all the c-units in a step $s \in Steps(Gs)$ is in the same step $t \in Steps(Gt)$. We should also prove that, for any $ct \in t, \phi^{-1}(ct) = cs$ belongs to s. The proof that we are going to give can be used also in this case. In fact, when we consider a $ct' \in t$ which we know to have a corresponding c-unit $cs' \in s$, we should prove that, if $\{ct, ct'\} \subseteq t, t \in Steps(Gt)$ and $\{cs, cs'\} \subseteq Gs$, such that $ct \overset{\Re_{Gt}(ct)}{\sim} cs$ and $ct' \overset{\Re_{Gt}(ct')}{\sim} cs'$, then cs, cs' belong to the same step, i.e. $Ig(cs) \equiv Ig(cs')$.

Let us show $Ig(ct_1) \equiv Ig(ct_2)$. Let us consider $\{X_1 = t_1\} \in Ig(ct_1)$. Then there exists $\{X = t\} \in Ig(cs_1) \equiv Ig(cs_2)$ such that $\{X = t\} \overset{\mathsf{proj}_1(\Re_{Gs}(cs_1))}{\sim} \{X_1 = t_1\}$. Also, for some $\{X_2 = t_2\} \in Ig(ct_2)$, $\{X = t\} \overset{\mathsf{proj}_1(\Re_{Gs}(cs_2))}{\sim} \{X_2 = t_2\}$.

By means of the existence of $\{X_2 = t_2\}$ we will prove the existence of $\{X_2' = t_2'\} \in Ig(ct_2)$ such that $\{X_1 = t_1\} \equiv \{X_2' = t_2'\}$.

In t, t_1, t_2 to any occurrence of a variable Z in t it corresponds occurrences of variables Z_1 in t_1 and Z_2 in t_2.

By hypothesis, $\Re_{Gs}(cs_1) = \Re_{Gs}(cs_2) = \Re_{Gt}(ct_1) = \Re_{Gt}(ct_2)$ and $\mathsf{proj}_1(\Re_{Gs}(cs_1)) = \mathsf{proj}_1(\Re_{Gs}(cs_2)) = \mathsf{proj}_1(\Re_{Gt}(ct_1)) = \mathsf{proj}_1(\Re_{Gt}(ct_2)) = \mathcal{R}$.

This means that, for an R-set $R \in \mathcal{R}$, it should be $\{X, X_1, X_2\} \subseteq R$. For any triple $\{Z, Z_1, Z_2\}$ there should also be an R-set $R \in \mathcal{R}$ such that $\{Z, Z_1, Z_2\} \subseteq \mathcal{R}$.

It is possible to prove that Z_1 and Z_2 belong to an R-set $\mathcal{R}$ iff at least one of the two occurs as eliminable variable in c-units $ct \sqsubseteq ct_1, ct' \sqsubseteq ct_2$.

As Z_1 is a parameter in ct_1, only Z_2 can occur as eliminable in $ct \sqsubseteq ct_1$. As Z_2 is a parameter in ct_2, we have $ct_2 \sqsubset ct$, by Lemma 3.

However, one of the two variables should appear as eliminable also in a c-unit $ct' \sqsubseteq ct_2$.

As Z_2 is a parameter in ct_2, it can only be Z_1 eliminable in ct'. As Z_1 is a parameter in ct_1, we have $ct_1 \sqsubset ct'$.

Then we have $ct_1 \sqsubset ct' \sqsubseteq ct_2 \sqsubset ct \sqsubseteq ct_1$, which is contradictory. The only alternative is $Z_1 \equiv Z_2$. Repeated application of this argument prove $t_1 \equiv t_2$.

For X_1 and X_2 it is possible to prove that it should be $\{X_2 = X_1\} \in Ig(ct_2)$ or ,for some term $t, \{X_2 = t, X_1 = t\} \in Ig(ct_2)$. The first possibility is excluded, because it would mean $\{X_1 = X_1\} \in Ig(ct_1)$. Then, when we selected the equation $\{X_2 = t_2\} \in Ig(ct_2)$, we could as well has selected $\{X_1 = t_1\} \in Ig(ct_2)$. The same reasonings are extended to all equations in $Ig(ct_1)$ and by symmetry, we have $Ig(ct_1) \equiv Ig(ct_2)$. □

Proof (Sketch) (Proposition 2) $Gs \approx Gt$ iff $Sim(Gs) \approx Sim(Gt)$. For symmetry, it is enough to prove $Sim(Gs) \subseteq Sim(Gt)$. Then, for any $\mathcal{S} \in Sim(Gs)$, it should exist $\mathcal{T} \in Sim(Gt)$ such that $\mathcal{S} \approx \mathcal{T}$. We will prove that $\mathcal{T} = [\phi(s_1), ..., \phi(s_m)]$ is such a simulation (the proof that $\mathcal{T}$ is indeed a simulation for Gt is not detailed here; it shows that, if $[\phi(s_1), ..., \phi(s_m)]$ is not a simulation, neither $[s_1, ..., s_m]$ is).

To prove $Sim(Gs) \approx Sim(Gt)$, we should prove, for $1 \leq j \leq m, soln(\bigcup_{i=1}^{j} s_i) = soln(\bigcup_{i=1}^{j} t_i)$ and $adm(\bigcup_{i=1}^{j} s_i) = adm(\bigcup_{i=1}^{j} t_i)$. By symmetry, it is enough to prove $soln(\bigcup_{i=1}^{j} s_i) \subseteq soln(\bigcup_{i=1}^{j} t_i)$ and $adm(\bigcup_{i=1}^{j} s_i) \subseteq adm(\bigcup_{i=1}^{j} t_i)$. The proof is by induction on the number of steps.

<u>Induction Base</u>

By hypothesis, $s_1 \overset{\Re_{Gs}(s_1)}{\sim} t_1$, where $\Re_{Gs}(s_1) = (\mathcal{R}_{Ig(s_1)}, \mathsf{lub}(\mathcal{R}_{Ig(s_1)}, \mathsf{lub}(\{\mathcal{R}_{Ou(cu)} : cu \in s_1\}))) = (\mathcal{R}_{Ig(t_1)}, \mathsf{lub}(\mathcal{R}_{Ig(t_1)}, \mathsf{lub}(\{\mathcal{R}_{Ou(cu)} : cu \in t_1\}))) = \Re_{Gs}(t_1)$, because both are first steps in a simulation.

<u>$soln(s_1) \subseteq soln(t_1)$.</u> Let us consider $\sigma \in soln(s_1)$. As $\mathcal{R}_{Ig(s_1)} = \mathcal{R}_{Ig(t_1)}$ and $Ig(s_1) \overset{\mathcal{R}_{Ig(s_1)}}{\sim} Ig(t_1)$ by Proposition 1 $Ig(s_1) \approx Ig(t_1)$ and $\sigma \in soln(Ig(t_1))$.

For any $\{X_t = u_t\} \equiv Ou(ct), ct \in t_1$, there exists $\{X_s = u_s\} \equiv Ou(cs)$ for some $cs \in s_1$ such that $\{X_t = u_t\} \overset{\mathrm{lub}(\mathcal{R}_{Ig(s_1)}, \mathrm{lub}(\{\mathcal{R}_{Ou(cu)} : cu \in s_1\}))}{\sim} \{X_s = u_s\}$. Then, corresponding occurrences of variables should belong to the same R-sets, and by Lemma 6 on the minimal R-class for an equations, it is easy to verify that they should receive the same value in σ.

Then $X_t\sigma = u_t\sigma$ holds. The same observations, extended to all $Ou(ct), ct \in t_1$, produce $\sigma \in soln(\{Ou(ct) : ct \in t_1\})$. As $\sigma \in soln(\{Ig(ct) : ct \in t_1\}), \sigma \in soln(t_1)$.

$\underline{adm(s_1) \subseteq adm(t_1)}$. Straightforward, because for $\mu \in adm(s_1), \Sigma(Ig(s_1)) \leq \mu \leq \sigma, \sigma \in soln(s_1)$. But $Ig(s_1) \approx Ig(t_1), \Sigma(Ig(s_1)) \approx \Sigma(Ig(t_1))$ and $soln(s_1) = soln(t_1)$. Then $\Sigma(Ig(t_1)) \leq \mu \leq \sigma, \sigma \in soln(t_1)$ and $\mu \in adm(t_1)$.

<u>Induction Step</u>

By inductive hypothesis, for $1 \leq j \leq k-1, soln(\bigcup_{i=1}^{j} s_i) \subseteq soln(\bigcup_{i=1}^{j} t_i)$ and $adm(\bigcup_{i=1}^{j} s_i) \subseteq adm(\bigcup_{i=1}^{j} t_i)$.

$\underline{soln(\bigcup_{i=1}^{k} s_i) \subseteq soln(\bigcup_{i=1}^{k} t_i)}$. $\sigma \in soln(\bigcup_{i=1}^{k} s_i)$ implies $\sigma \in soln(\bigcup_{i=1}^{k-1} s_i) \cap soln(s_k)$. Then,as $\sigma \in soln(\bigcup_{i=1}^{k-1} s_i), \sigma \in soln(\bigcup_{i=1}^{k-1} t_i)$, it is enough to prove $\sigma \in soln(t_k)$.

As $s_k \overset{\Re_{Gs}(s_k)}{\sim} t_k$, and $\Re_{Gs}(s_k) = \Re_{Gt}(t_k) = (\mathcal{R}_1, \mathcal{R}_2)$, we can extend the observations developed in the Induction Base for the output parts.

For any $\{X_t = u_t\} \in Ig(t_k)$, there exists $\{X_s = u_s\} \in Ig(s_k)$ such that $\{X_t = u_t\} \overset{\mathcal{R}_1}{\approx} \{X_s = u_s\}$.

Also,for any $\{X_t = u_t\} \in \{Ou(ct) : ct \in t_k\}$,there exists $\{X_s = u_s\} \in \{Ou(cs) : cs \in s_k\}$ such that $\{X_t = u_t\} \overset{\mathcal{R}_2}{\approx} \{X_s = u_s\}$.

Corresponding occurrences of variables should belong to the same R-sets, and it is possible to prove that they should receive the same value in any $\sigma \in soln(s_k)$. Then $\sigma \in soln(t_k)$.

$\underline{adm(\bigcup_{i=1}^{k} s_i) \subseteq adm(\bigcup_{i=1}^{k} t_i)}$. $\mu \in adm(\bigcup_{i=1}^{k} s_i)$ iff:

- $\mu_1 = \mu$ is IG-admissible for t_1;
- for $2 \leq i \leq k+1, \mu_i = \mathrm{lub}(\mu_{i-1}, \Sigma(s_{i-1}))$ is IG-admissible for s_i.

By inductive hypothesis, we need only to prove $\mu_k = \mathrm{lub}(\mu_{k-1}, \Sigma(s_{k-1}))$ is IG-admissible for t_k. μ_k is a candidate IG-admissible substitution for t_k because it is possible to prove that $\mathrm{lub}(\mu_{k-1}, \Sigma(s_{k-1})) \approx \mathrm{lub}(\mu, \mathrm{lub}(\{\Sigma(s_i) : i \leq k-1\})) \approx \mathrm{lub}(\mu, \mathrm{lub}(\{\Sigma(\phi(s_i)) : i \leq k-1\})) \approx \mathrm{lub}(\mu_{k-1}, \Sigma(t_{k-1}))$.

Now we should prove $\Sigma(Ig(t_k)) \leq \mu_k \leq \sigma$, for $\sigma \in soln(\bigcup_{i=1}^{k} t_i)$.

$\mu_k \leq \sigma$ is trivial, because $soln(\bigcup_{i=1}^{k} s_i) \subseteq soln(\bigcup_{i=1}^{k} t_i)$.

For $\Sigma(Ig(t_k)) \leq \mu_k$ we reason on $Ig(t_k)$. Without loss of generality, we can suppose $\Sigma(Ig(s_k)) = \{X_s \leftarrow u_s : \{X_s = u_s\} \in Ig(s_k)\}$ and $\Sigma(Ig(t_k)) = \{X_t \leftarrow u_t : \{X_t = u_t\} \in Ig(t_k)\}$. Then, if we prove that $\{X_t \leftarrow u_t\} \leq \mu_k$ for all $\{X_t = u_t\} \in Ig(t_k)$, we have indeed $\Sigma(Ig(t_k)) \leq \mu_k$.

For $Ig(t_k) \overset{\mathcal{R}_1}{\approx} Ig(s_k)$ we have that, for $\{X_t = u_t\} \in Ig(t_k)$, there exists $\{X_s = u_s\} \in Ig(s_k)$ such that $\{X_t = u_t\} \overset{\mathcal{R}_1}{\approx} \{X_s = u_s\}$.

A necessary condition for $\overset{\mathcal{R}_1}{\approx}$-equivalence is that to any variable occurrence in one equation corresponds in the other equation the occurrence of a variable which belongs to the same R-set (this is true also for $\{X_t, X_s\}$).

It is possible to prove (later we refer to this fact as (*)) that if $\{V_s, V_t\} \subseteq R, R \in \mathrm{proj}_1 \Re_{Gs}(cu)$, then one of the following is true:

[a] $\{V_s \leftarrow V_t\} \in \Sigma(\mathrm{lub}(\{cs : cs \sqsubset cu\}, Ig(cu))$ or $\{V_t \leftarrow V_s\} \in \Sigma(\mathrm{lub}(\{cs : cs \sqsubset cu\}, Ig(cu))$

[b] for a term r (eventually a variable Z), $\{V_s \leftarrow r, V_t \leftarrow r\} \subseteq \Sigma(\mathrm{lub}(\{cs : cs \sqsubset cu\}, Ig(cu))$.

Also, it is possible to prove that, as $\Sigma(\mathrm{lub}(\{cs : cs \sqsubset cu\}, Ig(cu)) \leq \mu_k$, the same hypothesis holds on μ_k.

We apply this result to $\{X_s, X_t\}$. When the rhs side of these substitutions for X_s and X_t are variables, the proof is particularly simple, because necessarily also u_s and u_t should be variables, let us suppose $u_s \equiv U_s$ and $u_t \equiv U_t$.

<u>Case.a</u> If $\{X_s \leftarrow X_t\} \in \mu_k$, and $\{X_s \leftarrow U_s\} \leq \mu_k$, we have that $\{U_s \leftarrow X_t\} \in \mu_k$. As U_s and U_t belong to the same R-set, we can only have $\{U_t \leftarrow X_t\}$, and $\{X_t \leftarrow U_t\} \leq \mu_k$.

If $\{X_t \leftarrow X_s\} \in \mu_k$, and $\{X_s \leftarrow U_s\} \leq \mu_k$, we have that $\{U_s \leftarrow X_s\} \in \mu_k$. For U_s and U_t we have $\{U_t \leftarrow X_s\}$, and $\{X_t \leftarrow U_t\} \leq \mu_k$.

<u>Case.b</u> If $r \equiv Z, \{X_s \leftarrow Z, X_t \leftarrow Z\} \in \mu_k$, and $\{X_s \leftarrow U_s\} \leq \mu_k$, we have that $\{U_s \leftarrow Z\} \in \mu_k$. For U_s and U_t we have $\{U_t \leftarrow Z\}$, and $\{X_t \leftarrow U_t\} \leq \mu_k$.

We are left with $\{X_s \leftarrow r, X_t \leftarrow r\} \subseteq \mu_k$ for a proper term r. We know $\{X_s \leftarrow u_s\} \leq \mu_k$.

If we see a term as a finite tree, u_t and r will differ only on some "leaves" of u_t which are labeled by variables.

To any occurrence of such variables U_t, it should correspond a variable U_s in u_s.

Let it be w the subterm in r originated at the position where U_t occurs in u_t.

Then it is possible to prove that, if $w \not\equiv U_s$, there should exist a substitution $\{U_s \leftarrow w\} \in \mu_k$ and, by (*), $\{U_t \leftarrow w\} \in \mu_k$. If $W \equiv U_s$, by (*), $\{U_t \leftarrow U_s\} \in \mu_k$.

When we compute $\{X_t \leftarrow u_t\}\mu_k$, these are the only substitutions which are effective on u_t and we have $\{X_t \leftarrow u_t\}\mu_k \equiv \{X_t \leftarrow u_t\mu_k\} \cup \{\mu_k - \{X_t \leftarrow r\}\} \equiv \{X_t \leftarrow r\}\} \cup \{\mu_k - \{X_t \leftarrow r\}\} \equiv \mu_k$ and $\{X_t \leftarrow u_t\} \leq \mu_k$. □