Title	Sufficient completeness of parameterized specifications in CafeOBJ
Author(s)	Nakamura, Masaki; Gaina, Daniel; Ogata, Kazuhiro; Futatsugi, Kokichi
Citation	Research report (School of Information Science, Japan Advanced Institute of Science and Technology), IS-RR-2014-005: 1-4
Issue Date	2014-12-15
Type	Technical Report
Text version	publisher
URL	http://hdl.handle.net/10119/12301
Rights	
Description	リサーチレポート（北陸先端科学技術大学院大学情報科学研究科）

IS-RR-2014-005

Sufficient completeness of parameterized specifications in CafeOBJ

Masaki Nakamura, Daniel Gaina, Kazuhiro Ogata, Kokichi Futatsugi

December 15, 2014

Sufficient completeness of parameterized specifications in CafeOBJ

Masaki Nakamura[1], Daniel Găină[2], Kazuhiro Ogata[2], and Kokichi Futatsugi[2]

[1] Toyama Prefectural University, 5180 Kurokawa, Imizu, Toyama, Japan
[2] Japan Advanced Institute of Science and Technology, 1-1 Asahidai, Nomi, Ishikawa, Japan

CafeOBJ is a specification language which supports several kinds of specifications [1] . In this study, we focus on constructor-based order-sorted (CBOS) equational specifications. A signature $(S, \leq, \Sigma, \Sigma^C)$ (abbr. Σ) consists of a set S of sorts, a poset $\leq$ on S, a S^+-sorted set Σ of operators, and a set $\Sigma^C \subseteq \Sigma$ of constructors. We use the notation for the complement set $\overline{\Sigma'} = \Sigma \setminus \Sigma'$, constrained sorts $S^{cs} = \{s \in S \mid f \in \Sigma^C_{ws} \vee (f \in \Sigma^C_{ws'} \wedge s' \leq s)\}$, loose sorts $S^{ls} = S \setminus S^{cs}$, and constrained operators $\Sigma^{S^{cs}} = \{f \in \Sigma_{ws} \mid w \in S^*, s \in S^{cs}\}$. A specification SP is a pair of a signature Σ and a set of equations on Σ. We use the subscript A_{SP} to refer the element of SP, e.g. S_{SP}, Σ_{SP}, E_{SP}, etc. Sufficient completeness is an important property which guarantees the existence of the initial model [3]. A sufficient condition of sufficient completeness is given in [2] as follows: $SP = ((S, \leq, \Sigma, \Sigma^C), E)$ is sufficiently complete if for each S^{ls}-sorted set Y of variables of loose sorts and each term $t \in T_{\Sigma^{S^{cs}}}(Y)$, there exists a term $u \in T_{\Sigma^C}(Y)$ such that $t =_E u$. We call the above condition SCE.

The theory of term rewriting systems (TRS) is useful to prove sufficient completeness, where equations are regarded as left-to-right rewrite rules. A term is E-reducible if it has a subterm which can be rewritten by some rewrite rule in E. It is known that the notion of basic terms is useful to show ground reducibility. We give a variant of basic terms for CBOS specifications.

Definition 1. For $\Sigma' \subseteq \Sigma$, $f(\bar{t})$ is a Σ'-basic if $f \in \overline{\Sigma'}$ and $\bar{t}$ are terms constructed from Σ' and loose variables Y.

Basic terms give us a sufficient condition of SCE. A specification SP satisfies SCE if SP is terminating, i.e, no infinite rewrite sequence exists, and all Σ^C-basic terms are E_{SP}-reducible [4]. We call the above condition SCR. The following $N+$ is a specification of natural number with the addition: $S_{N+} = \{Zero\ NzNat < Nat\}$, $\Sigma_{N+} = \{0 :\to Zero, s : Nat \to NzNat, \_+\_ : Nat\ Nat \to Nat\}$, $\Sigma^C_{N+} = \{0, s\}$, and $E_{N+} = \{X + 0 = X, X + s(Y) = s(X + Y)\}$, which is terminating and all Σ^C-basic terms $s^m(0) + s^n(0)$ are reducible, thus is sufficiently complete.

A parameterized specification is a specification morphism $i : P \to SP$ such that $i : \Sigma_P \to \Sigma_{SP}$ is an inclusion and $E_P \subseteq E_{SP}$. A view $v : P \to P'$ is a specification morphism from P to P', where $v(s) \in S_{P'}$ for $s \in S_P$, $v(f) \in \Sigma_{P'}$ for $f \in \Sigma_P$ and $v(e)$ is satisfied by P' for $e \in E_P$ where $v(e)$ is obtained by replacing each occurrence $f \in \Sigma_P$ in e with the operator $v(f)$. The instantiation of i by v, denoted by $SP(v)$, is obtained by constructing the pushout of $P' \leftarrow P \to SP$ [1]. Let E'_{SP} be $E_{SP} \setminus E_P$. Roughly speaking, $SP(v)$ is obtained by replacing P with P', and E'_{SP} with $\{v(e) \mid e \in E'_{SP}\}$. See [1] for more details. The following $i : FUN \to MAP$ is a parameterized specification of map functions on generic lists: $S_{FUN} = \{Elt\}$, $\Sigma_{FUN} = \{f : Elt \to Elt\}$ and $E_{FUN} = \emptyset$. $S_{MAP} = \{List\}$, $\Sigma_{MAP} = \{nil :\to List, \_;\_ : Elt\ List \to List, map : List \to List\}$, $\Sigma^C_{MAP} = \{nil, \_;\_\}$, and $E_{MAP} = \{map(nil) = nil, map(E; L) = f(E); map(L)\}$. Consider the specification $N{+}'$ obtained by adding $d(X) = X + X$ to $N{+}$ and the view $v_{fn} : FUN \to N{+}'$ where $v_{fn}(Elt) = Nat$ and $v_{fn}(f) = d$. Then, the instantiation $MAP(v_{fn})$ is a specification of lists on natural numbers where the function map takes $[n_0, n_1, \ldots, n_k]$ and returns $[2n_0, 2n_1, \ldots, 2n_k]$. $MAP(v_{fn})$ has the equations $\{map(nil) = nil,\ map(E; L) = d(E); map(L)\}$.

Given a parameterized specification $i : P \to SP$ and a view $v : P \to P'$, the challenge is to find sufficient conditions such that the instantiation $SP(v)$ is sufficient complete. MAP seems to be well-defined in the sense that after instantiation by a sufficiently complete specification, like $N{+}'$, the operator f becomes a constructor or an operator defined for all constructor terms, and thus $map(l)$ is equivalent to a constructor term for any $l \in T_{\Sigma^C}(Y)$. For example, in $MAP(v_{fn})$, $map(0; s(0); nil) \to^* 0 + 0; s(0) + s(0); nil \to^* 0; s(s(0)); nil \in T_{\Sigma^C}(\emptyset)$. However, MAP does not satisfy the above sufficient conditions SCE of sufficient completeness since a constrained term $f(X); nil$ does not have any equivalent constructor term. In order to cover such parameterized specifications, we generalize the condition SCR as follows:

Definition 2. A specification SP satisfies Σ'-SCR if SP is terminating and all Σ'-basic terms are E_{SP}-reducible.

Note that SCR is equivalent to Σ^C-SCR. We call $i : P \to SP$ left-P-free if the left-hand sides of the equations in E'_{SP} include no $f \in \Sigma_P$, constructor-preserving if for each $s \in S_P$, $T_{\Sigma^C}(Y)_s$ are same in both P and SP [4]. We have the following sufficient condition of sufficient completeness of instantiations.

Theorem 1. If (1) i is left-P-free and constructor-preserving, (2) SP satisfies $\Sigma^C_{SP} \cup \Sigma_P$-SCR, (3) P' satisfies SCR, (4) $SP(v)$ is terminating, then $SP(v)$ satisfies SCR.

Consider $i : FUN \to MAP$ and a view $v : FUN \to N{+}'$. i is left-FUN-free and constructor-preserving. All $\Sigma^C_{MAP} \cup \{f\}$-basic terms are in the form of either $map(nil)$ or $map(e; l)$, and reducible. $N{+}'$ satisfies SCR. Termination of $MAP(v)$ can be proved, for example, by the method in [4]. Thus, $MAP(v)$ satisfies SCR from Theorems 1 and is sufficiently complete.

References

1. Razvan Diaconescu and Kokichi Futatsugi. *CafeOBJ Report.* World Scientific, 1998.
2. Kokichi Futatsugi, Daniel Găină, and Kazuhiro Ogata. Principles of proof scores in CafeOBJ. *Theor. Comput. Sci.*, 464:90–112, December 2012.
3. Daniel Găină and Kokichi Futatsugi. Initial semantics in logics with constructors. *Journal of Logic and Computation*, 2012. doi: 10.1093/logcom/exs044.
4. Masaki Nakamura, et al. Incremental proofs of termination, confluence and sufficient completeness of OBJ specifications. LNCS 8373, pages 92–109. Springer, 2014.

Appendixes

A A generalization of SCE

We give a generalization of SCE, denoted by Σ'-SCE.

Definition 3. A specification SP satisfies Σ'-SCE if for each S^{ls}-sorted set Y of variables of loose sorts and each term $t \in T_{\Sigma S^{cs}}(Y)$, there exists a term $u \in T_{\Sigma'}(Y)$ such that $t =_E u$.

Note that Σ^C-SCE is equivalent to SCE. We have the following property.

Theorem 2. Σ'-SCR implies Σ'-SCE.

Proof. Let Y be a set of loose variables and $t \in T_{\Sigma S^{cs}}(Y)$. It suffices to show that $t \rightarrow^* u$ for some $u \in T_{\Sigma'}(Y)$. Note that $\rightarrow_E^* \subseteq =_E$. If $t \in T_{\Sigma'}(Y)$, then $u = t$. Assume t has an operator in $\overline{\Sigma'}$. Choose a subterms $f(\bar{t})$ whose root f is an operator in $\overline{\Sigma'}$ and $\bar{t}$ do not have any operator in $\overline{\Sigma'}$. $f(\bar{t})$ is a Σ'-basic term and reducible from the assumption of Σ'-SCR. Take t_1 as a term obtained by rewriting t, i.e., $t \rightarrow_E t_1$. If $t_1 \in T_{\Sigma'}(Y)$, then $u = t_1$. If not, repeat the same thing for t_1. From termination, there exists $t_n \in T_{\Sigma'}(Y)$ such that $t_i \rightarrow_E t_{i+1}$ $(i = 1, \ldots, n-1)$. Then $u = t_n$.

B Proof of Theorem 1

The notion of constructor-preserving has been defined for hierarchical extensions in [4], which can be modified for parameterized specifications straightforwardly.

Definition 4. [4] A parameterized specification $i : P \rightarrow SP$ is constructor-preserving if (1) for each $f \in (\Sigma^C_{SP})_{ws}$ such that $s \in S_P$, $f \in \Sigma_P$, and (2) for each $s \in S_P$, there is no $s' \in S_{SP} \setminus S_P$ such that $s' \leq_{SP} s$.

Note that if $i : P \rightarrow SP$ is constructor-preserving, $(T_{\Sigma^C_P}(Y))_s = (T_{\Sigma^C_{SP}}(Y))_s$ for each sort $s \in S_P$, and $(T_{\Sigma^C_{P'}}(Y))_s = (T_{\Sigma^C_{SP(v)}}(Y))_s$ for each view $v : P \rightarrow P'$ and sort $s \in S_{P'}$.

Theorem 1. If (1) i is left-P-free and constructor-preserving, (2) SP satisfies $\Sigma^C_{SP} \cup \Sigma_P$-SCR, (3) P' satisfies $\Sigma^C_{P'}$-SCR, (4) $SP(v)$ is terminating, then $SP(v)$ satisfies $\Sigma^C_{SP(v)}$-SCR.

Proof. From the assumption (4), it suffices to show that each $\Sigma^C_{SP(v)}$-basic term $f(\bar{t})$ is reducible.

- Consider the case of $f \in \Sigma_{P'}$. From the definition of basic terms, $f \in \overline{\Sigma^C_{SP(v)}} \subseteq \overline{\Sigma^C_{P'}}$ and each $t_i \in \{\bar{t}\}$ is in $(T_{\Sigma^C_{SP(v)}}(Y))_s$ for some $s \in S_{P'}$. From the assumption of constructor-preserving, $\bar{t} \in (T_{\Sigma^C_{P'}}(Y))_s$ and $f(\bar{t})$ is $E_{P'}$-reducible from (3). Since $E_{P'} \subseteq E_{SP(v)}$, $f(\bar{t})$ is also $E_{SP(v)}$-reducible.
- Consider the case of $f \in \Sigma_{SP(v)} \setminus \Sigma_{P'} = \Sigma_{SP} \setminus \Sigma_P$. Since $f(\bar{t})$ is $\Sigma^C_{SP(v)}$-basic, $f \notin \Sigma^C_{SP(v)}$ and $f \notin \Sigma^C_{SP}$. Thus $f \in \overline{\Sigma^C_{SP} \cup \Sigma_P}$. An argument term $t_i \in \{\bar{t}\}$ may have operators in $\Sigma_{P'}$. Make the term t'_i by replacing each maximal occurrence of $g \in (\Sigma_{P'})_{ws}$ in t_i with a fresh distinct variable $x \in X_s$. Note that $s \in S_{P'}$. Since t'_i is constructed from only $\Sigma_{SP \setminus P}$, there exists $t''_i \in T_{\Sigma^C_{SP}}(Y)$ where Y is a set

of loose variables. Thus, $f(\bar{t}'')$ is a $\Sigma^C_{SP} \cup \Sigma_P$-basic term and it is E_{SP}-reducible. Since $f \in \Sigma_{SP} \setminus \Sigma_P$, it is a redex of $E_{SP} \setminus E_P$, i.e. an instance of the left-hand side l of an equation in $E_{SP} \setminus E_P$. From the left-P-freeness, $f(\bar{t}')$ is also a redex of $E_{SP(v)} \setminus E_{P'}$. From the construction, $f(\bar{t})$ is an instance of $f(\bar{t}')$, and it is a redex of the same equation. Thus, it is $E_{SP(v)}$-reducible.

C Source codes

The following CafeOBJ codes correspond to the specifications FUN, $N+$, the parameterized specification $i : FUN \rightarrow MAP$ and the view $v_{fn} : FUN \rightarrow N+$.

```
mod* FUN{  [Elt] op f : Elt -> Elt }
mod! N+{ [Zero NzNat < Nat]
  op 0 : -> Zero  {constr}
  op s_ : Nat -> NzNat  {constr}
  op _+_ : Nat Nat -> Nat
  eq X:Nat + 0 = X .
  eq X:Nat + s Y:Nat = s (X + Y) . }
mod! MAP(Z :: FUN){   [List]
  op nil : -> List   {constr}
  op (_;_) : Elt List -> List  {constr}
  op map : List -> List
  eq map(nil) = nil .
  eq map(E:Elt ; L:List) = (f(E) ; map(L)) . }
view FN from FUN to N+ {
  sort Elt -> Nat,
  op f(E:Elt) -> (E:Nat + E) }
```

Note that CafeOBJ supports a view from operators to derived operators, like `op f(E) -> E + E` in `FN`. Such a view can be considered as the combination of an operator-to-operator view and a module where an extra equation $f(\bar{X}) = r$ is added, like `op f(E) -> d(E)` and $N + \cup\{d(E) = E + E\}$. The following is the result of the show command of the instantiation `MAP(FN)` in CafeOBJ system and the reduction command for `map(s 0 ; s s 0 ; nil)`:

```
CafeOBJ> show MAP(FN) .
module MAP(Z <= FN)
{
  imports {    protecting (N+)  }
  signature {    [ List ]
    op nil : -> List { constr prec: 0 }
    op _ ; _ : Nat List -> List { constr prec: 41 }
    op map : List -> List { prec: 0 }  }
  axioms {
    eq map(nil) = nil .
    eq map((E:Nat ; L:List)) = ((E + E) ; map(L)) .  }
}
CafeOBJ> red in MAP(FN) : map(s 0 ; s s 0 ; nil) .
-- reduce in MAP(Z <= FN) : (map(((s 0) ; ((s (s 0)) ; nil)))):List
((s (s 0)) ; ((s (s (s (s 0)))) ; nil)):List
(0.000 sec for parse, 8 rewrites(0.000 sec), 13 matches)
```