EFFECTIVE CHEMISTRY FOR SYNCHRONY AND ASYNCHRONY

Deepak Garg[1], Akash Lal[2], Sanjiva Prasad[3]

[1]*Carnegie Mellon University, Pittsburgh*
dg+@cs.cmu.edu

[2]*University of Wisconsin, Madison*
akash@cs.wisc.edu

[3]*Indian Institute of Technology Delhi, New Delhi*
sanjiva@cse.iitd.ernet.it

Abstract We study from an implementation viewpoint what constitutes a reasonable and effective notion of structural equivalence of terms in a calculus of concurrent processes and propose *operational effectiveness* criteria in the form of *confluence*, *coherence* and standardization properties on an oriented version of the structural laws. We revisit Berry and Boudol's Chemical Abstract Machine (CHAM) framework using operational effectiveness criteria. We illustrate our ideas with a new formulation of a CHAM for TCCS with external choice, one which is operationally effective unlike previous CHAM formulations, and demonstrate that the new CHAM is fully abstract with respect to the LTS semantics for TCCS. We then show how this approach extends to the synchronous calculus SCCS, for which a CHAM had hitherto not been proposed.

1. Introduction

Most presentations of structural operational semantics (SOS) of concurrent languages nowadays employ the notion of *structural equivalence* $\equiv$ between terms. This notion can be thought of as defining algebraic structure whereas transitions $\rightarrow$ modulo this equivalence represent computation across such structures. Typically included is a rule

$$\frac{P \equiv P' \qquad P \rightarrow Q \qquad Q \equiv Q'}{P' \rightarrow Q'} \ \textit{STRUCT}$$

which allows a term P to be readjusted into a form P' to which a specified transition rule applies. At an implementation level, however, such a rule is not "effective", in that it does not specify appropriate selections of elements within an equivalence class, nor does it bound the amount of "structural adjustment" to be performed. Indeed, there are few criteria for deciding what constitute reasonable notions of structural equivalence, beyond Milner's injunction [Mil93] that "structural laws should be digestible without concern for the dynamics of actions" and the obvious requirement that the notion be decidable. Relevant work on the latter issue

concerns decidability of structural equivalence for the π-calculus with the replication operator [EG01].

In this paper, we propose formal conditions that ensure reasonable and effective notions of structural equivalence. These *operational effectiveness* conditions, discussed in §3, are in the form of *confluence* and *coherence* properties for an oriented version of the structural rules, which yield a "standardization" result for execution sequences. Our conditions are closely related to the notions of coherence developed in the context of term rewriting modulo equivalence relations [Vir95]. Also related is Noll's work exploring the notion of coherence while expressing *finite* CCS (calculus of communicating systems) with SOS inference rules in a conditional rewriting framework [Nol99].

Operational effectiveness ensures correct and complete (abstract) implementations of the specified semantics. Taken together with termination of the oriented structural rules, these conditions yield a simple but complete implementation strategy. The import of these conditions is that they simplify establishing adequacy and/or full abstraction results. Confluence and coherence of oriented "administrative" transitions are also useful in analysis and verification of encodings of concurrent systems, since they vastly reduce the state space that needs exploration (see [AP98, GS95] for use of this idea). The connection with rewriting theory has additional benefits — while establishing confluence and coherence, standard rewriting techniques such as *completion* help ensure that there are "enough" structural rules.

We motivate, develop and present our ideas in the *chemical abstract machine* (CHAM) framework proposed by Berry and Boudol [BB90, BB92], which was the inspiration for Milner's formulation of the notion of structural equivalence in [Mil90]. The CHAM framework is an intuitive *style* of presenting operational semantics (presumably also an abstract implementation), where components of a parallel system are likened to *molecules*, and interaction between them is likened to a *chemical reaction* between ions. It exploits the commutative monoidal (*AC1*) properties of parallel composition to present systems as *solutions*, essentially finite multisets of molecules, within which *reactions* are specified locally in the form of conditional rewriting rules. "Structural adjustments", accomplished via the so-called *heating-cooling* and *clean-up rules* and permutations on molecules in a solution via a *magical mixing* mechanism (which can be seen as a prototypical treatment of mobility), allow distant components to react, thus dismantling the bureaucratic rigidity imposed by syntax. We argue that operational effectiveness provides an important criterion for assessing a CHAM specification, and for realizing an abstract implementation from it. It is central in our articulation of a new perspective on the CHAM framework, namely that the essence of CHAMs is that they define operationally effective rewrite systems modulo an *AC* equational theory. We must clarify that though our presentation is in the CHAM framework, the notion of operational effectiveness applies to any style of specification based on rewriting.

Our alternative perspective on the CHAM framework makes possible using a uniform disciplined CHAM idiom for expressing constructs involving non-local interaction, such as external choice, which are problematic in an asynchronous system with purely local interaction [NP96, Pal97]. Several distributed systems and protocols employ non-local interaction via some infrastructure or exhibit some degree of synchrony with the environment, and we believe that it is important for a robust framework for specifying concurrent behaviour to be able to express such *mediated* or *catalyzed* interaction *at an appropriate level of abstraction*. Roughly speaking, our alternative formulation trades the simplicity of autarkic asynchronous computation for applicability of the CHAM idea to more complex interactions. The confluence and coherence conditions provide the necessary discipline for structuring computation and controlling the effects of non-local interaction. We illustrate this idea by focussing on concurrency combinators such as external choice in a variant of CCS and synchronous parallel composition in Milner's synchronous calculus of communicating systems (SCCS) [Mil83].

Organization of the Paper. In §2 we introduce the CHAM framework and our notation. The idea of operational effectiveness for CHAMs is described in §3. The original CHAM in [BB90, BB92] for the variant of CCS called TCCS [NH87] fails to satisfy the confluence-coherence properties, and accordingly we present a reworked CHAM for TCCS in §4. We believe that this CHAM also suffices for the π-calculus, at least the part without name-matching, since it implements scope extrusion and the other structural equivalences. Further, we show that this CHAM is in full agreement with the standard labelled transition system (LTS) semantics for TCCS, with bisimilarity as the notion of equivalence. This result improves on the full abstraction result sketched by Boudol [Bou94] in that it works for external choice contexts as well. The proof technique we use seems to be widely applicable, and relies on the confluence and coherence properties of structural rules.
We then explore a "chemistry for synchrony" in §5, providing a CHAM for a version of SCCS. This is the first synchronous CHAM of which we are aware (hitherto all CHAMs were for calculi with asynchronous process execution). Paucity of space prevents us from presenting here a treatment of choice in SCCS, which appears in the full version of this paper. The proof of correctness follows the same template as that for the TCCS CHAM. The main analysis required in all our example CHAMs involves showing that various rules commute. The TCCS and SCCS CHAM examples use mechanisms based on information-carrying tags for capturing non-local interactions in external choice and synchronous parallel composition. These tags are, however, manipulated by local rules, giving workable implementations of these constructs that involve non-local interaction. We believe that this model can be extended to distributed settings because of its compositional nature. The concluding section (§6) comments on the essence of this alternative view on CHAMs, extensions and future directions of work. The full version of this paper can be obtained from `http://www.cse.iitd.ac.in/~sanjiva`.

2. Preliminaries

A CHAM consists of a specification of its *molecules*, *solutions* and *transformation rules* on solutions. The rewriting semantics is that a rule $l \rightarrow r$ may be applied to any solution that contains substitution instances of the molecules of l, which are replaced by the instances under the same substitution of the molecules in r. Consider the following sub-language (called CCS$^-$) of CCS, where p denotes a typical process term and α a typical action defined over a set $Act = \mathcal{N} \cup \overline{\mathcal{N}}$ where $\mathcal{N}$ is a set of names and $\overline{\mathcal{N}} = \{\overline{x} \mid x \in \mathcal{N}\}$ is the set of "co-names". The "co" operation is involutive, *i.e.*, $\overline{\overline{x}} = x$.

$$p ::= 0 \mid \alpha.p \mid \nu x.p \mid p|p \mid \ \ldots$$

Here 0 stands for inaction, "." denotes action prefixing, "|" parallel composition and "ν" the restriction operation (written in the notation favoured in the π-calculus).
A CHAM *solution*, typically denoted $\mathcal{S}$ or $\{\!|m_1, \ldots, m_k|\!\}$, consists of a finite multiset of *molecules* m_i, delimited by the membrane brackets "$\{\!|$ $|\!\}$". In the CHAM framework, *all* transformations are specified on solutions. *Molecules* are basically terms, extended to allow solutions within them, and certain constructions on solutions. Molecules for CCS$^-$ (typically m) are defined as follows:

$$m ::= p \mid \nu x.\mathcal{S} \qquad \mathcal{S} ::= \{\!|m_1, \ldots, m_k|\!\} \ \ (k \geq 0)$$

Let $\uplus$ denote multiset union on solutions, *i.e.*,

$$\{\!|m_1, \ldots, m_k|\!\} \uplus \{\!|m'_1, \ldots, m'_l|\!\} = \{\!|m_1, \ldots, m_k, m'_1, \ldots, m'_l|\!\}.$$

Rules peculiar to a calculus consist of: (a) *Reaction rules*, presented as conditional rewrite rules on solutions, which are of the form $\{\!|m_1, \ldots, m_k|\!\} \mapsto \{\!|m'_1, \ldots, m'_l|\!\}$. These basic computational steps, which may be non-deterministic, are denoted using the arrow "$\mapsto$", possibly sub-

scripted by a rule label. (b) *Structural rules*, which are either the *reversible* "heating-cooling" rules $\mathcal{S} \rightleftharpoons \mathcal{S}'$ or oriented "clean-up" rules $\mathcal{S} \rightsquigarrow \mathcal{S}'$ that get rid of inert terms. Heating rules usually are of the form $\{\!|m|\!\} \rightharpoonup \{\!|m'_1, \ldots m'_k|\!\}$ and intuitively are intended to prepare a solution for reaction. Cooling rules, the inverses of the heating ones, usually are of the form $\{\!|m'_1, \ldots m'_k|\!\} \rightharpoondown \{\!|m|\!\}$. We use the symbol $\rightharpoondown$ to denote $\rightharpoonup^{-1}$, the symmetric inverse of the heating relation. Clean-up transitions will be denoted by $\rightsquigarrow$. Clean-up rules are distinguished, as a matter of taste, from heating rules in that they do not increase the ability of a solution to react, and their orientation is obvious.

Laws common to a variety of calculi include: (a) The *Chemical Law*, which permits rewriting within a solution according to a *locality principle* that allows reactions and adjustments to occur independently of the other molecules in a solution (here "$\rightarrow$" denotes any rewriting, whether reaction or structural):

$$\frac{\mathcal{S} \;\rightarrow\; \mathcal{S}'}{\mathcal{S} \uplus \mathcal{S}'' \;\rightarrow\; \mathcal{S}' \uplus \mathcal{S}''}$$

(b) The *Membrane Law*, which permits reactions to occur within reduction contexts:

$$\frac{\mathcal{S} \;\rightarrow\; \mathcal{S}'}{\{\!|C[\mathcal{S}]|\!\} \;\rightarrow\; \{\!|C[\mathcal{S}']|\!\}}$$

A reduction context $C[\ ]$ is a molecule with a *solution*-shaped hole in it, that is only a solution may be placed in such a hole. Berry and Boudol also employed an *Airlock Law* in some CHAMs, particularly for implementing choice. It allows particles to be isolated from a solution, to support restricted interaction with the external context: $\{\!|m, m_1, \cdots, m_n|\!\} \rightleftharpoons \{\!|m \triangleleft \{\!|m_1, \cdots, m_n|\!\}|\!\}$. In our treatment, we drop such a law, since it does not have the desired confluence properties.

Notation. The symbol $\rightarrow$ will be used to denote the union of (*i.e.*, any of) the relations $\rightharpoonup$, $\rightharpoondown$, $\rightsquigarrow$, and $\mapsto$, whereas the symbol $\rightarrow_{\mathcal{A}}$ abbreviates $\rightharpoonup \cup \rightsquigarrow$. The symbol $\leftrightarrow_{\mathcal{A}}$ denotes the symmetric closure of $\rightarrow_{\mathcal{A}}$, and $=_{\mathcal{A}}$ its reflexive-transitive-symmetric closure. For any relation R, let $R^{\leftrightarrow}$, R^{+} and R^{*} denote its symmetric, its transitive, and its reflexive-transitive closures, respectively. The different rewrite rules are labelled, and we will often subscript these reaction/heating/cooling/clean-up relations with the labels of the rules of interest.

The operational rules for CCS^{-} are specified as follows. Communication is specified through the irreversible reaction rule (schema): $\quad (R) \quad \{\!|x.p, \overline{x}.q|\!\} \mapsto \{\!|p, q|\!\}$

For the CCS^{-} subset the structural rules are:

$$\begin{array}{llll}
(P) & \{\!|p|q|\!\} \rightleftharpoons \{\!|p, q|\!\} & (0c) & \{\!|0|\!\} \rightsquigarrow \{\!|\ |\!\} \\
(M) & \{\!|\nu x.p|\!\} \rightleftharpoons \{\!|\nu x.\{\!|p|\!\}|\!\} & (\nu c) & \{\!|\nu x.\mathcal{S}|\!\} \rightsquigarrow \mathcal{S} \quad x \notin fv(\mathcal{S}) \\
(E) & \{\!|\nu x.S, p|\!\} \rightleftharpoons \{\!|\nu x.(\{\!|p|\!\} \uplus S)|\!\} \quad x \notin fv(p) & &
\end{array}$$

All these rules are applicable whenever permitted by the Chemical and Membrane laws. The contexts to be considered are:

$$C ::= [\ ] \mid C \uplus \mathcal{S} \mid \{\!|\nu x.C|\!\}$$

The structural rules are adapted from the CHAM given by Boudol for the π-calculus [Bou94], rather than the TCCS CHAM given in [BB92]. The main difference is that the airlock mechanism is not used and instead the rule (E) is introduced which allows scope extrusion. *Note that in this specification we identify terms upto α-renaming of bound variables and swapping of consecutive restriction membranes*. The rules apply modulo an equational theory $\mathcal{E}$ (on solutions) induced by the following equalities (here M denotes a molecule or a solution):

$$\begin{array}{ll}
(\alpha{-}cnv) & \nu x.M \;=\; \nu y.M[y/x] \;\; (y \notin fv(M)) \\
(\nu{-}swap) & \nu x.\{\!|\nu y.\mathcal{S}|\!\} \;=\; \nu y.\{\!|\nu x.\mathcal{S}|\!\}
\end{array}$$

The first equality expresses the essence of what is meant by a term with bound variables. We will outline (in §3) how the second equality can be treated by oriented rewriting modulo an *AC*

theory.

Proposition 1. For any two CCS$^-$ terms p and q, $p \equiv q$ iff $\{\!|p|\!\} =_{\mathcal{A}} \{\!|q|\!\}$, where $\equiv$ denotes the standard notion of structural equivalence on CCS$^-$ terms.

3. Effective Structural Transformations

The above-mentioned intuitions for the heating-cooling and clean-up rules suggest that the rules should be *oriented* (rightwards) in the direction of heating and clean-up. If these oriented $\rightarrow_{\mathcal{A}}$-moves are confluent, they may be applied in any order and (if terminating) yield unique normal forms, which are more reactive than all other structurally equivalent forms. Note that confluence of heating implies the following commutation, which allows cooling to be postponed: $\rightharpoonup^*; \rightharpoonup^* \subseteq \rightharpoonup^*; \rightharpoondown^*$. Further, if heating/clean-up moves are not to prune away a potential reaction, then a series of heating/clean-up steps and any reaction step can commute. Combining these intuitions, we arrive at the following definition of *operational effectiveness*.

Definition 2 (Operational Effectiveness). A set of oriented structural rules (heating and clean-up rules of a CHAM) is said to be *operationally effective* if the following two properties hold:

1. ($\rightarrow_{\mathcal{A}}$-confluence) The relation $\rightarrow_{\mathcal{A}}$ is Church-Rosser: if $\mathcal{S} \rightarrow^*_{\mathcal{A}} \mathcal{S}_1$ and $\mathcal{S} \rightarrow^*_{\mathcal{A}} \mathcal{S}_2$, then $\mathcal{S}_1 \rightarrow^*_{\mathcal{A}} \mathcal{S}_3$ and $\mathcal{S}_2 \rightarrow^*_{\mathcal{A}} \mathcal{S}_3$ for some $\mathcal{S}_3$.
2. ($\rightarrow^*_{\mathcal{A}} - \mapsto$-commutation or coherence) For all $\mathcal{S}, \mathcal{S}_1, \mathcal{S}_2$ if $\mathcal{S} \rightarrow^*_{\mathcal{A}} \mathcal{S}_1$ and $\mathcal{S} \mapsto \mathcal{S}_2$, then there exists $\mathcal{S}_3$ such that $\mathcal{S}_1 \rightarrow^*_{\mathcal{A}} \mapsto \mathcal{S}_3$ and $\mathcal{S}_2 \rightarrow^*_{\mathcal{A}} \mathcal{S}_3$.

The *coherence* condition may seem more general than needed for many instances, but even in our example CCS$^-$ CHAM, several suggested stronger versions are unable to handle adequately, *e.g.*, extrusion of the scope of a restriction by the (E) rule. *Confluence* is an essential requirement since without it, coherence is ineffective.
Immediate consequences of operational effectiveness are $=_{\mathcal{A}} \subseteq \rightarrow^*_{\mathcal{A}}; (\rightarrow^{-1}_{\mathcal{A}})^*$ and $(\rightarrow^{-1}_{\mathcal{A}})^*; \mapsto \quad \subseteq \quad \rightarrow^*_{\mathcal{A}}; \mapsto; (\rightarrow^{-1}_{\mathcal{A}})^*$, from which follows a standardization for reduction sequences.

Theorem 3 (Standardization). If a set of oriented structural rules is operationally effective then $\forall n \geq 0, \quad (=_{\mathcal{A}}; \mapsto; =_{\mathcal{A}})^n \subseteq \rightarrow^*_{\mathcal{A}}; (\mapsto; \rightarrow^*_{\mathcal{A}})^n; (\rightarrow^{-1}_{\mathcal{A}})^*$

Proposition 4. The CHAM for CCS$^-$ is operationally effective.

This result follows from showing that the various rules commute as required. In fact, most pairs of structural rules commute strongly (strong diamond property), with the exception of the (E) rule.

We propose operational effectiveness as an important criterion for assessing a CHAM specification. We observe that some CHAMs in the literature (for the π-calculus and the Join calculus [FG96]) seem reasonable (effective), whereas the TCCS and the γ-calculus CHAMs in [BB90, BB92] are not, since the laws for restriction and the airlock law in the first, and the hatching and membrane laws in the second, lead to non-confluent heating.

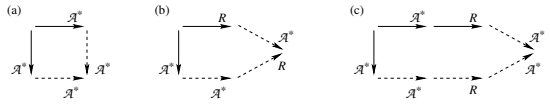
Termination and strategy for implementation. In the CCS$^-$ CHAM given above, we can show that the administrative $\rightarrow_{\mathcal{A}}$-moves are strongly normalizing. The nontrivial aspect here involves treating extrusion of the scope of restriction by the rule (E).

Proposition 5 (Strong Normalization). The relations $\rightarrow_{\mathcal{A}}$ (heating+clean-up) and $\rightharpoondown$ (cooling) are strongly normalizing in the CHAM for CCS$^-$.

Standardization and termination yield a fairly simple but complete implementation strategy (even with guarded recursion): heat/clean-up a solution as much as possible using rules other than the recursion rule, then unfold *once* each recursive term, and then heat/clean-up as much as possible using the other rules.

3.1 Rewriting modulo equivalence

A crucial question is whether it is semantically correct to work with reduction (reaction) modulo oriented structural rules instead of reduction modulo structural equivalence. Our confluence-coherence conditions ensure that it is indeed so for effective CHAMs. We later found that in [Vir95], Viry has studied this issue in the general setting of *oriented rewrite theories* (ORTs).
Technically, an equational theory is decomposed into a "core" notion of equality $\mathcal{E}$ (with respect to which matching is tractable) and a collection of oriented rewrite rules $\mathcal{A}$. The question is "when can $\mathcal{R}$ modulo $\mathcal{E} \cup \mathcal{A}^{\leftrightarrow}$ (the semantics) be simulated by $(\mathcal{A})^* \mathcal{R} (\mathcal{A})^*$ modulo $\mathcal{E}$ (the implementation)?" For CHAMs, $\mathcal{E}$ should capture only the essential equational theory for solutions. Viry has identified *coherence conditions* which suffice to establish that the implementation relations are *complete* with respect to the specified semantics (the other direction, *soundness*, is trivial). In the figure below, we depict (with solid lines quantified universally and dotted lines existentially) the following properties: (a) confluence of the oriented $\mathcal{A}$ rules, (b) our coherence property, and (c) Viry's *strong coherence* property.



Our coherence property implies Viry's strong coherence property, and therefore, ensures completeness of the implementation with respect to the semantics. Indeed, it does not need the extra $\mathcal{A}^*$ moves after the $\mathcal{R}$ move on the lower (existential) branch to complete the diagram. This stronger property slightly simplifies implementation and reasoning, *e.g.*, the proofs of full abstraction. Note that coherence does not require $\rightarrow_{\mathcal{A}}$, to be terminating (modulo $\mathcal{E}$) as, for instance, Noll does in his formalization of CCS [Nol99].
Next, we argue that the essence of an effective CHAM *is that it defines an operationally effective rewrite system modulo an AC equational theory*. Rewriting modulo arbitrary equational theories may not be tractable [Nol99]. Viry has given conditions for checking the coherence conditions (under the assumption that $\mathcal{A}$ is terminating), by checking a finite number of critical pairs modulo $\mathcal{E}$ when an instance of a *generic permutation lemma* holds for $\mathcal{E}$. Instances of this lemma are known for $\mathcal{E} = \emptyset$, $\mathcal{E} = A$ (associativity) and $\mathcal{E} = AC$ (associativity and commutativity). The last of these theories agrees well with the notion of multiset rewriting which is at the core of the CHAM framework. In the sequel, we illustrate that the salient concurrency combinators — restriction, external choice and synchronous parallel composition — can all be treated within an *AC* framework.
The identity laws for parallel composition and non-deterministic choice, and idempotence for the latter combinator are treated as rules in $\mathcal{A}$, namely as clean-up rules in the CHAM. The conditions on these clean-up rules ensure that there are no problematic critical pairs, thus avoiding the termination and completion related problems that may arise in rewriting modulo identity. Idempotence is dealt with a fairly simple commutation argument.
If we disregard α-conversion, which is uncontroversial and can anyway be treated in a first-order theory using an indexing scheme, the only equality that is not an *AC* property is the (*ν-swap)* equation introduced §1 for the CCS$^-$ CHAM. This was necessary to ensure commutation when using the (E) rule. It can be eliminated by the following mildly different treatment of restriction, one that highlights that restriction in some sense satisfies *AC* properties (based on those of

set union).

Factoring the (E) rule. Instead of molecules of the form $\nu x.\mathcal{S}$, we will write instead $\mathcal{S}_X$, where X is a *set* of names, which are considered bound in $\mathcal{S}$. The (M) rule is recast as $\{\!|\nu x.p|\!\} \rightleftharpoons \{\!|\{\!|p|\!\}_{\{x\}}|\!\}$. Now the (E) rule can now be factored into the two rules:

$$\{\!|\mathcal{S}_X, p|\!\} \rightleftharpoons_{EM} \{\!|(\mathcal{S} \uplus \{\!|p|\!\})_X|\!\} \;\; X \cap fv(p) = \emptyset$$

$$\{\!|\mathcal{S}_X, \mathcal{S}'_Y|\!\} \rightleftharpoons_{EF} \{\!|(\mathcal{S} \uplus \mathcal{S}')_{X+Y}|\!\} \;\; X \cap fv(\mathcal{S}') = Y \cap fv(\mathcal{S}) = \emptyset$$

With this factoring, the strong normalization property is preserved, and stronger statements can be made regarding rule commutation — (EF) commutes strongly, whereas (EM) commutes, but possibly weakly, with other rules.

3.2 Agreement with LTS semantics

What is the relationship between a calculus $\mathcal{P}$ equipped with a labelled transition system (LTS) $\rightarrow^{\alpha}$, and its purported CHAM formulation $\mathcal{C}$? We outline the key notions and a template for proving the correctness of the CHAMs in §4 and §5. First, we equip $\mathcal{C}$ with a *labelled transition relation* $\mapsto^{\alpha}$ based on a suitable notion of observability for solutions. Let $\Longrightarrow^{\alpha} = \rightarrow^{*}_{\mathcal{A}}; \mapsto^{\alpha}; \rightarrow^{*}_{\mathcal{A}}$. This will be the transition relation on $\mathcal{C}$.

Definition 6. Given a LTS $\langle \mathcal{G}, \rightarrow^{\square} \rangle$, a symmetric relation $\mathcal{R} \subseteq \mathcal{G} \times \mathcal{G}$ is called a bisimulation on $\mathcal{G}$ if whenever $(p, q) \in \mathcal{R}$ and $p \rightarrow^{\alpha} p'$, there exists $q' \in \mathcal{G}$ such that $q \rightarrow^{\alpha} q'$ and $(p', q') \in \mathcal{R}$.

Suppose $\sim_p$ and $\sim_c$ stand for bisimulation equivalence in $\mathcal{P}$ and $\mathcal{C}$ respectively. A translation $\langle . \rangle : \mathcal{P} \rightarrow \mathcal{C}$ is called *adequate* (sound) if $\langle p \rangle \sim_c \langle q \rangle$ implies $p \sim_p q$, and *fully abstract* if $p \sim_p q$ implies $\langle p \rangle \sim_c \langle q \rangle$ as well. (These properties also apply to reasonable notions of equivalence other than bisimulation.)

Lemma 7. If $\rightarrow_{\mathcal{A}}$-moves of a CHAM are operationally effective, then $=_{\mathcal{A}}$ is a bisimulation on CHAM configurations.

The crucial fact used here is that due to coherence, the (eventual) possibility of a reaction is preserved across $\rightarrow_{\mathcal{A}}$-moves.

Definition 8 (Forward and Backward Simulation). CHAM $\mathcal{C}$ *forward simulates* $\mathcal{P}$ if for all process $p, q \in \mathcal{P}$ such that $p \rightarrow^{\alpha} q$, there exists a CHAM configuration $\mathcal{S}$ such that $\langle p \rangle \Longrightarrow^{\alpha} \mathcal{S} =_{\mathcal{A}} \langle q \rangle$. CHAM $\mathcal{C}$ *backward simulates* $\mathcal{P}$ if for any process p and CHAM configuration $\mathcal{S}$, such that $\langle p \rangle \Longrightarrow^{\alpha} \mathcal{S}$, there is a process p' such that $\langle p' \rangle =_{\mathcal{A}} \mathcal{S}$ and $p \rightarrow^{\alpha} p'$.

Theorem 9 (Bisimulation). Let CHAM $\mathcal{C}$ be operationally effective and both forward and backward simulate $\mathcal{P}$. Then the following two relations are bisimulations on $\mathcal{C}$-configurations and processes in $\mathcal{P}$ respectively.

1 $\mathcal{B}_c = \{(\mathcal{S}_1, \mathcal{S}_2) \,|\exists p, q \; s.t. \; p \sim_p q, \mathcal{S}_1 =_{\mathcal{A}} \langle p \rangle, \mathcal{S}_2 =_{\mathcal{A}} \langle q \rangle\}$

2 $\mathcal{B}_p = \{(p, q) \,|\exists \mathcal{S}_1, \mathcal{S}_2 \; s.t. \; \langle p \rangle =_{\mathcal{A}} \mathcal{S}_1, \langle q \rangle =_{\mathcal{A}} \mathcal{S}_2, \mathcal{S}_1 \sim_c \mathcal{S}_2\}$

The forward and backward simulation conditions relate the $\rightarrow^{\alpha}$ moves of a process and the $\Longrightarrow^{\alpha}$-moves of its image under the translation. Lemma 7 allows us to find a suitable $\Longrightarrow^{\alpha}$-derivative of a given CHAM configuration to fulfil the bisimilarity requirements.

Corollary 10 (Full Abstraction). If the conditions of Theorem 9 hold for a CHAM, then $\langle p \rangle \sim_c \langle q \rangle$ if and only if $p \sim_p q$.

4. An Effective CHAM for TCCS

We now extend CCS$^-$ with external choice, and guarded recursion, *i.e.*, in $\mathit{fix}_i(\vec{x} = \vec{p})$, every occurrence of a process variable x_i in the p_i's is within an action-prefixed term. Guardedness is a vital condition for providing an effective treatment in the presence of recursion.

$$p \ ::= \ \ldots \mid p[]p \mid \mathit{fix}_i(\vec{x} = \vec{p})$$

The standard semantics for TCCS is:

$$\frac{}{\alpha.p \ \mapsto^{\alpha} \ p} \qquad \frac{p_1 \mapsto^{\alpha} p_1' \quad p_2 \mapsto^{\overline{\alpha}} p_2'}{p_1|p_2 \ \mapsto \ p_1'|p_2'} \qquad \frac{p_1 \ \mapsto \ p_1'}{p_1|p_2 \ \mapsto \ p_1'|p_2} \qquad \frac{p_2 \ \mapsto \ p_2'}{p_1|p_2 \ \mapsto \ p_1|p_2'}$$

$$\frac{p_1 \mapsto^{\alpha} p_1'}{p_1|p_2 \ \mapsto^{\alpha} \ p_1'|p_2} \qquad \frac{p_2 \mapsto^{\alpha} p_2'}{p_1|p_2 \ \mapsto^{\alpha} \ p_1|p_2'} \qquad \frac{p \ \mapsto \ p'}{\nu x.p \ \mapsto \ \nu x.p'} \qquad \frac{p \mapsto^{\alpha} p' \quad (\alpha \notin \{x, \overline{x}\})}{\nu x.p \ \mapsto^{\alpha} \ \nu x.p'}$$

$$\frac{p_1 \ \mapsto \ p_1'}{p_1[]p_2 \ \mapsto \ p_1'[]p_2} \qquad \frac{p_2 \ \mapsto \ p_2'}{p_1[]p_2 \ \mapsto \ p_1[]p_2'} \qquad \frac{p_1 \mapsto^{\alpha} p_1'}{p_1[]p_2 \mapsto^{\alpha} p_1'} \qquad \frac{p_2 \mapsto^{\alpha} p_2'}{p_1[]p_2 \mapsto^{\alpha} p_2'}$$

$$\frac{}{\mathit{fix}_i(\vec{x} = \vec{p}) \ \mapsto \ p_i\lceil \mathit{fix}_j(\vec{x} = \vec{p}) \ / \ x_j \rceil_{j=1}^{n}}$$

In [BB90, BB92], external choice was implemented using reversible rules for airlocks and *heavy ions*, which carry tags (l or r) memo-ing the component of a choice from which an action originated. Once choice is resolved by the context, irreversible projection rules eliminate the other alternatives. This treatment is quite awkward: it introduces a great deal of new syntax, and is rigid in tagging the heavy choice ions, contrary to the *AC* properties of choice. Furthermore the airlock law leads to non-confluence.

We generalize the tagging approach to a compositional treatment, while presenting an effective CHAM rewrite system. TCCS external choice is implemented using "speculative concurrent execution", *i.e.*, running the various (tagged) alternatives concurrently, until one is selected. Thereupon the others are culled away, using the tags to determine which components to retain or kill. Given any two sub-processes in a TCCS term, they are either in exclusive choice or in parallel with each other. We use an *unordered, finitely branching tree* of nodes alternatingly marked $\mathbf{P}$ and $\mathbf{C}$ (for parallel composition and choice respectively) to represent such relations between processes. The leaves are marked with the tags. This abstract data structure, called an *exclusion tree*, serves as a catalyst that mediates the non-local interaction necessary for external choice. While we present it as a "global" component, its manipulation may admit some parallelism. Let $\mathcal{L} = \{a_1, a_2, \ldots\}$ be an infinite set of labels. An exclusion tree, denoted by T, is defined by the following grammar.

$$T \ ::= \ T_p \qquad T_p \ ::= \ a \mid \mathbf{P}(T_c, \ldots, T_c) \qquad T_c \ ::= \ a \mid \mathbf{C}(T_p, \ldots, T_p)$$

$\mathbf{P}$ and $\mathbf{C}$ denote internal nodes of the tree with finite non-zero arity. Given an exclusion tree T, arbitrary nodes and subtrees rooted at those nodes are denoted by n and its decorated variants. Given a node n, the *type* of n is $\mathbf{P}$ (respectively $\mathbf{C}$) if the subtree rooted at n was produced from the non-terminal T_p (resp. T_c). Contexts for exclusion trees are denoted by C_T.

$$C_T \ ::= \ C_p \qquad C_p \ ::= \ [\,] \mid \mathbf{P}(C_c, T_c, \ldots, T_c) \qquad C_c \ ::= \ [\,] \mid \mathbf{C}(C_p, T_p, \ldots, T_p)$$

CHAM configurations are pairs, written $T \vdash \mathcal{S}$, where T is an exclusion tree and $\mathcal{S}$ is a solution. It is assumed that all leaves in T are distinct; this property is preserved by the rewriting rules. TCCS terms in the solution $\mathcal{S}$ are labeled from the set $\mathcal{L}$. Molecules may now be redefined.

$$m \ ::= \ p^a \mid \nu x.\mathcal{S}$$

A molecule p^a is termed *active* if its tag a is a leaf in the tree T. *Only active molecules are allowed to take part in a reaction.* The rest may be garbage collected.

The equational theory $\mathcal{E}$ on configurations $T \vdash \mathcal{S}$ is obtained by "lifting" to configurations the equational theory $\mathcal{E}$ defined earlier for solutions, and adding the equations

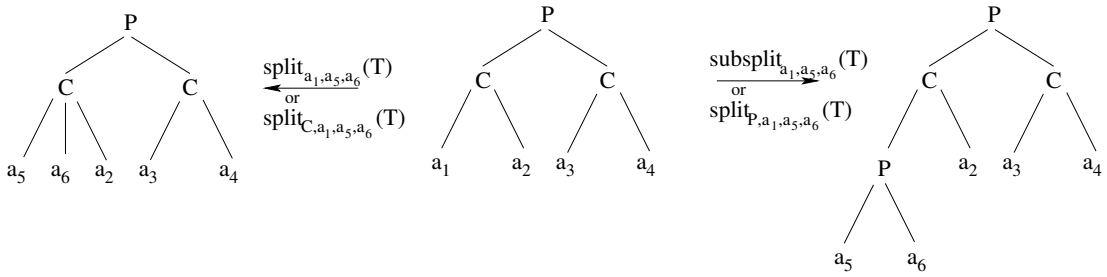


Figure 1. The *split* and *subsplit* operations

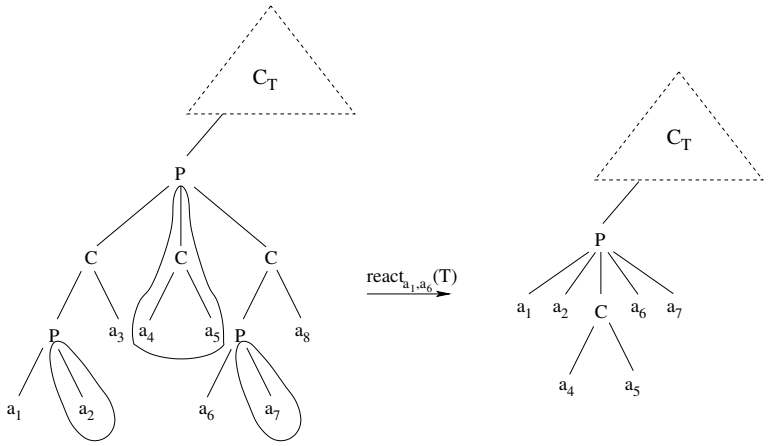


Figure 2. The *react* operation

$$\begin{aligned}\mathbf{P}(\ldots,T_i,\ldots,T_j,\ldots) &= \mathbf{P}(\ldots,T_j,\ldots,T_i,\ldots)\\ \mathbf{C}(\ldots,T_i,\ldots,T_j,\ldots) &= \mathbf{C}(\ldots,T_j,\ldots,T_i,\ldots)\end{aligned}$$

to emphasize the unordered (*AC*) nature of the tree. In addition, $\mathcal{E}$ also contains the following renaming equality for labels.

$$T \vdash \mathcal{S} = T[a_2/a_1] \vdash \mathcal{S}[a_2/a_1] \quad a_2 \notin T, \mathcal{S}$$

Let C denote a *solution context* with a "solution-shaped" hole.

$$C ::= [\,] \mid C \uplus \mathcal{S} \mid \{\!|\nu x.C|\!\}$$

We say that $a \in T$ if the tree T has a leaf labeled a. Also, $(n_1 \leftarrow n_2) \in T$ if the parent of node n_1 in T is the node n_2. If n is a node with only one child n_1, $remove_n(T)$ is the tree T with the nodes n, n_1 removed and the children of n_1 connected to $parent(n)$; if n_1 is itself a leaf, it is connected to $parent(n)$. $T - (n_1 \leftarrow n_2)$ denotes tree T with the node n_1 and all its children removed provided $(n_1 \leftarrow n_2) \in T$. If $n_2 \in T$, $T + (n_1 \leftarrow n_2)$ is the tree T with the additional subtree n_1 added below n_2. Define $split_{a,a_1,a_2}(T) = ((T - (a \leftarrow n)) + (a_1 \leftarrow n)) + (a_2 \leftarrow n)$ when $(a \leftarrow n) \in T$ and $a_1, a_2 \in \mathcal{L}$ are not in T. Also, $subsplit_{a,a_1,a_2}(T) = ((T - (a \leftarrow n)) + (n' \leftarrow n)) + (a_1 \leftarrow n') + (a_2 \leftarrow n')$ if $(a \leftarrow n) \in T$. n' is a new internal node having a type different from n and $a_1, a_2 \in \mathcal{L}$ are labels not in T. Figure 1 depicts these operations graphically.

For $t \in \{\mathbf{P}, \mathbf{C}\}$ define

$$T \vdash C[\{\!|(p|q)^a|\!\}] \rightleftharpoons_P split_{\mathbf{P},a,a_1,a_2}(T) \vdash C[\{\!|p^{a_1}, q^{a_2}|\!\}] \qquad (a \in T)$$

$$T \vdash C[\{\!|(\nu x.p)^a|\!\}] \rightleftharpoons_M T \vdash C[\{\!|\nu x.\{\!|p^a|\!\}|\!\}] \qquad (a \in T)$$

$$T \vdash C[\{\!|m, \nu x.\mathcal{S}|\!\}] \rightleftharpoons_E T \vdash C[\{\!|\nu x.(\mathcal{S} \uplus \{\!|m|\!\})|\!\}] \qquad (x \notin fv(m))$$

$$T \vdash C[\{\!|(p[]q)^a|\!\}] \rightleftharpoons_C split_{\mathbf{C},a,a_1,a_2}(T) \vdash C[\{\!|p^{a_1}, q^{a_2}|\!\}] \qquad (a \in T)$$

$$T \vdash C[\{\!|\mathit{fix}_i(\vec{x} = \vec{p})^a|\!\}] \rightleftharpoons_F T \vdash C[\{\!|(p_i[\mathit{fix}_j(\vec{x} = \vec{p}) \,/\, x_j]_{j=1}^n)^a|\!\}] \qquad (a \in T)$$

$$T \vdash C[\{\!|p^a|\!\}] \rightsquigarrow_{gc} T \vdash C[\{\!|\,|\!\}] \qquad \text{if } a \notin \mathbf{T}$$

$$T \vdash C[\{\!|\nu x.\mathcal{S}|\!\}] \rightsquigarrow_{\nu c} T \vdash C[\mathcal{S}] \qquad \text{if } x \notin fv(\mathcal{S})$$

$$T \vdash C[\{\!|0^a|\!\}] \rightsquigarrow_{0c} T - (a \leftarrow n) \vdash C[\{\!|\,|\!\}] \qquad (a \leftarrow n) \in T$$

$$T + (n_1 \leftarrow n) \vdash \mathcal{S} \rightsquigarrow_{aL} remove_n(T) \vdash \mathcal{S} \qquad \text{if } n \text{ has only one child } n_1 \text{ in } T$$

$$T \vdash C[\{\!|(\alpha.p)^{a_1}, (\overline{\alpha}.q)^{a_2}|\!\}] \mapsto_R react_{a_1,a_2}(T) \vdash C[\{\!|p^{a_1}, q^{a_2}|\!\}] \qquad \text{if } a_1, a_2 \in T \text{ and } react_{a_1,a_2}(T) \text{ is defined}$$

Figure 3. Rules for the TCCS CHAM.

$$split_{t,a,a_1,a_2}(T) = \begin{cases} split_{a,a_1,a_2}(T) & \text{if } t = \mathit{type}(\mathit{parent}(a)) \\ subsplit_{a,a_1,a_2}(T) & otherwise \end{cases}$$

Let $LCA_T(a_1, a_2)$ be the least common ancestor of the leaves a_1 and a_2 in T. Let $a_1, a_2 \in T$ and $n = LCA_T(a_1, a_2)$. Let n have type $\mathbf{P}$ and $T = C_T[T']$ where T' is the subtree of T rooted at n and C_T is the rest of the tree T. Define $react_{a_1,a_2}(T)$ to be the tree $C_T[\mathbf{P}(a_1, a_2, T_1, T_2, \ldots)]$ where $T_1, T_2, \ldots$ are all the subtrees of T whose roots were children of nodes of type $\mathbf{P}$ occurring on the unique paths from a_1 to n and a_2 to n in T. The $react$ operation is depicted graphically in Figure 2, with the subtrees $T_1, T_2, \ldots$ circled.

The rewriting rules for TCCS CHAM configurations are given in Figure 3. The rules are presented as context-embedded rewrites, rather than specifying elementary rewrites and inductively propagating these steps via Chemical and Membrane laws. This is just a matter of technical convenience. Such an approach is often used for specifying reduction semantics for λ-calculi instead of inference rules for induction cases.

The heating-cooling rules $(M), (E)$ are as before, except for the tag management. The (P) rule splits the leaf corresponding to the term where it is applied. The (C) rule deals with external choice and allows a term $p[]q$ to decompose into p and q tagged with leaves occurring as separate children of a $\mathbf{C}$ node. The rule (F) for fix-points allows recursive definitions to be unfolded in the heating direction, and is standard. The (gc) clean-up rule allows one to "garbage collect" *inactive* molecules, those whose tags are not in T. The (aL) clean-up rule removes nodes that represent a singleton term in a choice or parallel context. The rules $(0c)$ and (νc) are as before. The proviso on the reaction rule ensures that both reagent molecules are active, and not in mutual exclusion. The reaction eliminates from the exclusion tree all tags that marked terms mutually exclusive of either reacting molecule.

We define the administrative moves of the TCCS CHAM as the heating and cleanup rules: $\rightarrow_{\mathcal{A}} = \rightharpoonup \cup \rightsquigarrow$. With a small extension of the earlier treatment, it is not difficult to show that $\rightarrow_{\mathcal{A}}$ is strongly normalizing (since recursion is guarded, the use of $\rightharpoonup_F$ is bounded).

Definition 11 (LTS). A labelled transition relation $\mapsto^{\alpha}$ can be defined as:
$T \vdash C[\{\!|(\alpha.p)^a|\!\}] \mapsto^{\alpha} T' \vdash C[\{\!|p^a|\!\}]$ if $a \in T$ and $C[\,]$ does not restrict α. Here $T' = \mathbf{P}(a, T_1, T_2, \ldots)$ where $T_1, T_2, \ldots$ are all the subtrees of T whose root is a child of a

node of type $\mathbf{P}$ occurring on the path from a to the root of T.

Definition 12 (Translation). For p in TCCS, define $\langle p \rangle = a \vdash \{\!|p^a|\!\} \quad a \in \mathcal{L}$

Lemma 13 (TCCS-Administrative Moves). The administrative moves $\rightarrow_{\mathcal{A}}$ of TCCS CHAM are operationally effective for both the LTS ($\mapsto^{\alpha}$) and reduction ($\mapsto_R$).

Lemma 14 (TCCS-Simulation). The TCCS CHAM satisfies the properties of forward and backward simulation with respect to both the LTS and the reduction semantics.

The proof of this lemma employs an alternative formulation of the CHAM, which is closer to the inductive style followed in LTS semantics of TCCS. In fact, the CHAM we have presented was systematically derived from the equivalent alternative "inductive" LTS presentation. That formulation was first "closed" with contexts to yield a reduction system with inductive laws, and then "flattened" with respect to contexts to yield the present rewrite-rule form.

Theorem 15 (Standardization and Full Abstraction).

1 For the TCCS CHAM, $(=_{\mathcal{A}}; \rightarrow^{\alpha}; =_{\mathcal{A}})^n \subseteq \rightarrow_{\mathcal{A}}^*; (\rightarrow^{\alpha}; \rightarrow_{\mathcal{A}}^*)^n; (\rightarrow_{\mathcal{A}}^{-1})^*$

2 For the TCCS CHAM, $(=_{\mathcal{A}}; \mapsto_R; =_{\mathcal{A}})^n \subseteq \rightarrow_{\mathcal{A}}^*; (\mapsto_R; \rightarrow_{\mathcal{A}}^*)^n; (\rightarrow_{\mathcal{A}}^{-1})^*$

3 The TCCS CHAM is a fully abstract implementation.

We note in passing that in [Vir95], Viry had specified LOTOS semantics (which is closely related to CCS) as an oriented rewriting theory. We believe that his formulation is somewhat unsatisfactory since it includes the "Expansion Theorem" [Mil89] in the oriented structural rules $\mathcal{A}$. This amounts to embedding a *particular* notion of observation and program equivalence into the structural equivalence, which seems to run contrary to Milner's injunction on keeping structural equivalence independent of the dynamics.

5. An Effective CHAM for SCCS

We now consider a variant of SCCS [Mil83], a calculus in which process execution is *synchronous*. Assume that the set of actions Act forms an Abelian monoid, under the operation $\cdot$, with 1 denoting the identity element. Let $\alpha \in Act$, a set of actions, and let $X \subseteq Act$. For brevity, we write $\alpha\beta$ for $\alpha \cdot \beta$.
The syntax of the subset of SCCS we consider is given by the abstract grammar:

$$p \ ::= \ 0 \mid \alpha.p \mid p_1|p_2 \mid \nu X.p \mid \mathit{fix}_i(\vec{x} = \vec{p})$$

0 represents inability to execute, "." denotes prefixing, and "|" is now synchronous parallel composition. Due to space restrictions, in this paper, we omit the choice operator considered by Milner in his original presentation of SCCS. Further we assume that we have only guarded recursion. We must clarify that for continuity with the previous section, we employ a restriction operator νX similar in spirit to that in CCS. Our $\nu X.p$ can be defined as $p \wr (Act - X)$ in Milner's syntax.
The LTS semantics for this SCCS subset are:

$$\frac{}{\alpha.p \ \mapsto^{\alpha} \ p} \qquad \frac{p_1 \mapsto^{\alpha_1} p_1' \quad p_2 \mapsto^{\alpha_2} p_2'}{p_1|p_2 \ \mapsto^{\alpha_1\alpha_2} \ p_1'|p_2'}$$

$$\frac{p \ \mapsto^{\alpha} \ p'}{\nu X.p \ \mapsto^{\alpha} \ \nu X.p'} \quad \alpha \notin X \qquad \frac{(p_i[\mathit{fix}_j(\vec{x} = \vec{p}) \ / \ x_j]_{j=1}^n) \ \mapsto^{\alpha} \ p'}{\mathit{fix}_i(\vec{x} = \vec{p}) \ \mapsto^{\alpha} \ p'}$$

In a synchronous calculus, all processes act in concert. At first blush, this suggests an alternative chemical law of the form:

$$\frac{\mathcal{S}_1 \rightarrow \mathcal{S}_1' \qquad \mathcal{S}_2 \rightarrow \mathcal{S}_2'}{\mathcal{S}_1 \uplus \mathcal{S}_2 \rightarrow \mathcal{S}_1' \uplus \mathcal{S}_2'}$$

In a CHAM, however, the processes must be structurally adjusted to be ready for synchronizing with one another. Since the number of administrative $\rightarrow_{\mathcal{A}}$ moves can vary for different components, they cannot be performed in lockstep. Thus we continue with the old chemical law, at least for the structural rules, though we discuss below an alternative chemical law for the synchronization steps.

Since all terms in a SCCS process act together to produce a composite action, their individual actions need to be propagated upward on the structure of the term, and only at the top level is it decided whether an action can take place. Our CHAM implementation mimics this idea, but several implementation-level rewrite steps are needed to accomplish a semantic transition. We use tags to propagate actions. Let $\circ$ be an element not in Act (we call this element *"Notag"*, and it is used to mark molecules prior to ionization or after action propagation). Tags are elements of $Act \cup \{\circ\}$. We denote tags by the letter t and its decorated variants.

Molecules and solutions. Solutions (denoted by $\mathcal{S}$) and molecules (denoted by m) are defined by the following grammar.

$$m \ ::= \ p \mid \nu X.\mathcal{S}^t \qquad \mathcal{S} \ ::= \ \{\!|m_1^{t_1}, \ldots, m_n^{t_n}|\!\} \quad n \geq 0$$

Tagged solutions and tagged molecules are solutions and molecules with a tag on them. The tag is written as a superscript on the solution or molecule. We define contexts for (untagged) solutions by the following grammar.

$$C \ ::= \ [\,]^t \mid (\{\!|(\nu X.C)^{t_1}|\!\} \uplus \mathcal{S})^{t_2}$$

Rules. The heating/cooling rules given below can be freely applied wherever permitted by a Chemical Law or Membrane Law. The (I) rule describes *ionization* of a prefixed term.

$$\{\!|(\alpha.p)^\circ|\!\} \rightleftharpoons_I \{\!|p^\alpha|\!\} \qquad \{\!|(\nu X.p)^\circ|\!\} \rightleftharpoons_M \{\!|(\nu X.\{\!|p^\circ|\!\}^\circ)^\circ|\!\}$$
$$\{\!|(p|q)^\circ|\!\} \rightleftharpoons_P \{\!|p^\circ, q^\circ|\!\} \qquad \{\!|fix_i(\vec{x} = \vec{p})^\circ|\!\} \rightleftharpoons_F \{\!|(p_i[fix_j(\vec{x} = \vec{p}) \,/\, x_j]_{j=1}^n)^\circ|\!\}$$

Reaction. Reaction in SCCS is a LTS move $\mathcal{S}^\alpha \ \mapsto_R^\alpha \ \mathcal{S}^\circ$. Reaction is a *top level* rewrite, to which the Membrane Law and Chemical Law do *not* apply.

Propagation Rules. The propagation rules given below propagate actions on molecules and solutions upwards on the structure of the system. Synchronisation is facilitated by the rule (PU). The rule (νU) allows actions to be propagated past a restriction.

$$\{\!|m_1^{\alpha_1}, \ldots, m_n^{\alpha_n}|\!\}^\circ \ \hookrightarrow_{PU} \ \{\!|m_1^\circ, \ldots, m_n^\circ|\!\}^{\alpha_1 \cdots \alpha_n}$$
$$\{\!|(\nu X.\mathcal{S}^\alpha)^\circ|\!\} \ \hookrightarrow_{\nu U} \ \{\!|(\nu X.\mathcal{S}^\circ)^\alpha|\!\} \quad \alpha \notin X$$

Both the chemical and membrane law may be used in conjunction with (νU). Observe that the rule (PU) works on *tagged* solutions. The usual chemical law does not apply to this rule. However the membrane law does. Contexts for tagged solutions are defined as follows.

$$C_t \ ::= \ [\,] \mid (\{\!|(\nu X.C_t)^{t_1}|\!\} \uplus S)^{t_2}$$

Alternative Chemical Law for (PU). As the (PU) rule is essentially about synchronizing actions from different components, the usual chemical law does not apply. However, the following alternative chemical law achieves (piecemeal) the synchronization of actions and upward propagation done by (PU):

$$\{\!|m^\alpha|\!\}^\circ \ \hookrightarrow_{PU} \ \{\!|m^\circ|\!\}^\alpha \qquad \frac{\mathcal{S}_1^\circ \ \hookrightarrow_{PU} \ (\mathcal{S}_1')^{\alpha_1} \quad \mathcal{S}_2^\circ \ \hookrightarrow_{PU} \ (\mathcal{S}_2')^{\alpha_2}}{(\mathcal{S}_1 \uplus \mathcal{S}_2)^\circ \ \hookrightarrow_{PU} \ (\mathcal{S}_1' \uplus \mathcal{S}_2')^{\alpha_1 \cdot \alpha_2}}$$

Results. The SCCS CHAM given here is also operationally effective, and is in agreement with its LTS semantics. This supports our case that synchronous operations can be dealt with in

a disciplined CHAM framework.

Definition 16 (Administrative Moves). For the SCCS CHAM, $\rightarrow_{\mathcal{A}} = \rightharpoonup \cup \hookrightarrow$.

Definition 17 (Translation). For a SCCS process p, we define $\langle p \rangle = \{\!|p^{\circ}|\!\}^{\circ}$.

Lemma 18 (SCCS-Administrative Moves). The administrative moves $\rightarrow_{\mathcal{A}}$ of the SCCS CHAM are operationally effective for the LTS $(\mapsto_R^{\alpha})$.

Lemma 19 (SCCS Simulation). The SCCS CHAM forward and backward simulates the LTS semantics of SCCS.

Theorem 20 (Standardization and Full Abstraction).

1 For the SCCS CHAM, $(=_{\mathcal{A}}; \mapsto_R^{\alpha}; =_{\mathcal{A}})^n \subseteq \rightarrow_{\mathcal{A}}^*; (\mapsto_R^{\alpha}; \rightarrow_{\mathcal{A}}^*)^n; (\rightarrow_{\mathcal{A}}^{-1})^*$

2 The SCCS CHAM is a fully abstract implementation.

6. Conclusion

We have argued that operational effectiveness is an important criterion for assessing any structural congruence or CHAM specification, since it ensures that the implementation is reasonable and in agreement with the intended semantics. The critical notions are those of confluence and coherence, which turn out to be valuable tools for reasoning about systems and in proofs of adequacy and full abstraction. We believe that the CHAM framework is worth extending beyond asynchronous systems to accommodate non-local interactions and (partial) synchronous operators. Accordingly, we have proposed an alternative "artificial chemistry" in which reactions are "mediated", and in which operational effectiveness provides a vital discipline. Indeed, we contend that the *locality principle* articulated by Banâtre, Boudol and others should relate not merely to the particular Chemical Law they presented (which works well for asynchronous systems) but to these notions of confluence and coherence, which are at the heart of any reasonable CHAM treatment.

In our two examples, we have considered limited subsets of TCCS and SCCS to illustrate the ideas, and for establishing standardization and full abstraction. This is not a serious limitation. For instance, internal choice can be treated by adding the following *reaction* rules. Since internal choice leads to non-confluent behaviour, it should not be a structural rule. These rules do not affect the properties of operational effectiveness.

$$T \vdash \{\!|(p_1 \oplus p_2)^a|\!\} \ \mapsto_{IC} \ T \vdash \{\!|p_i^a|\!\} \quad \text{if } a \in T \quad i \in \{1,2\}$$

τ actions are invisible moves which resolve internal choice but not external choice. They can be treated by adding an extra reaction rule.

$$T \vdash \{\!|(\tau.p)^a|\!\} \ \mapsto_R \ T \vdash \{\!|p^a|\!\} \quad \text{if } a \in T$$

It is intuitive and satisfying that reasonable notions of structural equivalence arise from rule commutations. We believe that structural equivalences arise from permitted commutations in a framework such as rewriting logic. Indeed, the semantic foundations of CHAMs in conditional rewriting logic deserve greater study ([Mes92] shows how CHAMs can be expressed in that framework, though properties such as coherence of those rewriting rules have not been studied further there or in subsequent related work, *e.g.*, [VM00]). We also feel that rewriting logic can provide a framework for exploring the connections between asynchronous and synchronous calculi, since it can express both kinds of chemistry.

In summary, we tacitly identify the essential mechanism of a CHAM as being oriented rewriting modulo a collection of *AC* equational theories. We may posit that the essential aspects of good

CHAM formulations are: (a) The structural rules are factored into a "core" *AC* equality theory and an orientable set of rewrite rules. (b) The oriented structural rules $\rightarrow_{\mathcal{A}}$ satisfy commutation properties, thus exhibiting confluence and strong coherence of $\rightarrow_{\mathcal{A}}$ with $\mapsto$. (c) Establishing strong coherence is kept relatively simple by avoiding problematic critical pairs, particularly in non-superposition cases, *e.g.*, by disallowing reactions within molecules that are heatable.

Acknowledgement. This work was supported in part by MHRD projects RP01425 and RP01432 and a grant from SUN Microsystems.

References

[AP98] R. Amadio and S. Prasad. Modelling IP mobility. In *Proceedings of CONCUR'98*, *LNCS* vol. 146: 301–316. Springer, 1998.

[BB90] G. Berry and G. Boudol. The chemical abstract machine. In *Proceedings of PoPL'90*, pages 81–94. ACM, 1990.

[BB92] G. Berry and G. Boudol. The chemical abstract machine. *TCS*, 96:217–248, 1992.

[Bou94] G. Boudol. Some chemical abstract machines. In *A Decade of Concurrency*, *LNCS* vol. 803: 92–123. Springer, 1994.

[EG01] J. Engelfriet and T. Gelsema. Structural inclusion in the pi-calculus with replication. *TCS*, 258(1-2):131–168, 2001.

[FG96] C. Fournet and G. Gonthier. The reflexive chemical abstract machine and the join-calculus. In *Proceedings of PoPL'96*, pages 372–385. ACM, 1996.

[GS95] J. F. Groote and J. Springintveld. Focus points and convergent process operators. Logic Group Preprint Series 142, Department of Philosophy, Utrecht University, 1995.

[Mes92] J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *TCS*, 96(1):73–155, 1992.

[Mil83] R. Milner. Calculi for synchrony and asynchrony. *TCS*, 25:267–310, 1983.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice Hall International, 1989.

[Mil90] R. Milner. Functions as processes. In *Proceedings of ICALP'90*, *LNCS* vol. 443: 167–180. Springer-Verlag, 1990.

[Mil93] R. Milner. The polyadic π-calculus: A tutorial. In W. Brauer, F.L. Bauer, and H. Schwichtenberg, eds, *Logic and Algebra of Specification*. Springer, 1993.

[NH87] R. De Nicola and M. Hennessy. CCS without τ's. In *Proceedings of TAPSOFT'87*, *LNCS* vol. 249: 138–152. Springer, 1987.

[Nol99] T. Noll. On coherence properties in term rewriting models of concurrency. In *Proceedings of CONCUR'99*, *LNCS* vol. 1664: 478–493, Springer, 1999.

[NP96] U. Nestmann and B. C. Pierce. Decoding choice encodings. In *Proceedings of CONCUR'96*, *LNCS* vol. 1119: 179–194. Springer, 1996.

[Pal97] C Palamidessi. Comparing the expressive power of the synchronous and the asynchronous pi-calculus. In *Proceedings of PoPL'97*, pages 256–265. ACM, 1997.

[VM00] A. Verdejo and N. Martí-Oliet. Implementing CCS in Maude. In *Proceedings of FORTE 2000*, pages 351–366, Kluwer, 2000.

[Vir95] P. Viry. Rewriting modulo a rewrite system. Technical Report TR-95-20, Dipartimento di Informatica, Univ. Pisa, Dec 1995.