LFCS

Laboratory for Foundations of Computer Science
Department of Computer Science - University of Edinburgh

Characteristic Formulae for CCS with Divergence

by

Bernhard Steffen

LFCS Report Series

ECS-LFCS-89-76
(also published as CSR-294-89)

May 1989

LFCS
Department of Computer Science
University of Edinburgh
The King's Buildings
Edinburgh EH9 3JZ

Characteristic Formulae for CCS with Divergence

Bernhard Steffen
Laboratory for Foundations of Computer Science
University of Edinburgh
GB Edinburgh EH9 3JZ

Abstract

Characteristic formulae have been introduced by Graf and Sifakis to relate equational reasoning about processes to reasoning in a modal logic, and therefore to allow proofs about processes to be carried out in a logical framework. Based upon an intuitionistic understanding of the modal mu-calculus, this paper extends the results of Graf and Sifakis in two respects. First, it covers not only finite processes, but finite state processes. Second, it handles not only bisimulation–like equivalences but also preorders, which are sensitive to liveness properties [1].

1 Motivation

Transition systems are often used as models for concurrent programs. By themselves they are too concrete to provide an appropriate notion of computational equivalence or simulation. Addition of an explicit equivalence or preorder relation on transition systems overcomes this problem, by identifying processes (nodes) of 'similar' computational power. In this paper we investigate transition systems which explicitly provide divergence information [19] together with a notion of simulation given by a divergence sensitive preorder. (Think of divergence as the potential for infinite internal chatter.) As the main result we show how to obtain *characteristic formulae*[2] for this notion of simulation within an intuitionistic interpretation [19] of a sublanguage of Kozen's modal mu–calculus [11]. Application to CCS [14, 16] delivers characteristic formulae for bisimulation equivalences, congruences and preorders. Therefore this paper extends the results of Graf and Sifakis [9] in two respects. First, it covers not only finite processes, but finite state processes. Second, it handles not only bisimulation–like equivalences (which are also dealt with in [2, 10]) but also *preorders* [21], which are sensitive to liveness

[1] This paper is an extended version of [18].

[2] Characteristic formulae are the equivalent logical reformulation of an equational process specification [9].

properties. — The results of this paper are the basis for an extension of the Concurrency Workbench[3] to automatically derive logical specifications from process specifications.

2 Extended Transition Systems

An *extended finite state transition system* T is a quadruple $(\mathbf{P}, \rightarrow, Act, \Uparrow)$ where

1. $\mathbf{P}$ is a finite set (of *processes* or *states*);
2. Act is a finite set (of *observable actions*);
3. $\rightarrow$ is a mapping which associates, with each $a \in Act$, a relation $\xrightarrow{a} \subseteq \mathbf{P} \times \mathbf{P}$;
4. $\Uparrow$ is a mapping, which associates with each $\mu \in Act \cup \{\varepsilon\}$, where $\varepsilon \notin Act$ represents unobservable actions, a unary predicate $\Uparrow\mu$ on $\mathbf{P}$. — Instead of $\Uparrow\mu(p)$ we write $p \Uparrow \mu$, and we require that $p \Uparrow \varepsilon$ implies $p \Uparrow a$ for every observable action $a \in Act$.

Typically $\mathbf{P}$ is a set of program states, and for each $a \in Act$ the relationship $p \xrightarrow{a} q$ indicates that p can evolve to q under the observation of a. The fourth component $\Uparrow$ is introduced to express *divergence* potential: $p \Uparrow \mu$ is intended to mean that p can be triggered by means of an offer of a μ-action to evolve autonomously for ever without responding to the environment. The predicate $\Uparrow\varepsilon$ does not depend on any observable action. Therefore $\Uparrow\varepsilon$ implies $\Uparrow a$, for every $a \in Act$. That is the reason for calling $\Uparrow\varepsilon$ the *global divergence* predicate. — The negation of $p \Uparrow \mu$, μ-convergence, is written $p \Downarrow \mu$, and we will represent states or processes (of a transition system) as *rooted transition systems* (i.e. as pairs (T, p_0), where p_0 is the *start state*).

Typical examples in CCS would be[4]:

1) $fix(X, \tau.X + a.nil)$ expressed as $(\,(\{p_0, p_1\},\ p_0 \xrightarrow{a} p_1,\ \{a\},\ p_0 \Uparrow \varepsilon),\ p_0\,)$
2) $a.nil$ expressed as $(\,(\{p_0, p_1\},\ p_0 \xrightarrow{a} p_1,\ \{a\}, -\,),\ p_0\,)$

These two processes are *observationally equivalent* in the sense of [14, 16]

$$fix(X, \tau.X + a.nil) \approx a.nil$$

even though the first process can engage in an infinite internal chatter, and therefore refuse to response to an a-action at all. This motivates the definition of a *preorder*, $\preceq$, between processes:

[3] The Concurrency Workbench is an automated tool for the analysis of concurrent systems [5, 6, 7].

[4] In CCS divergence usually is interpreted as the potential for an infinite τ-sequence.

Definition 2.1 $\preceq$ *is the union of all relations* R *satisfying that* pRq *implies*

1. $\forall a \forall p'.\ if\ p \xrightarrow{a} p'\ then\ \exists q'.\ q \xrightarrow{a} q'\ and\ p'Rq'$
2. $\forall a.\ if\ p \Downarrow a\ then\ (q \Downarrow a\ and\ \forall q'.\ if\ q \xrightarrow{a} q'\ then\ \exists p'.p \xrightarrow{a} p'\ and\ p'Rq')$

The preorder $\preceq$, which is in fact the greatest relation satisfying the two conditions mentioned above, now distinguishes between the two example processes. Indeed, we still have:

$$fix(X, \tau.X + a.nil) \ \preceq\ a.nil$$

but the converse fails:

$$a.nil \ \not\preceq\ fix(X, \tau.X + a.nil)$$

3 The Logic and its Characterization of $\preceq$

Our modal logic $\mathcal{L}$ is essentially a sublanguage of Kozen's modal *μ-calculus* [11]. It involves *modalities* for expressing transitional change and a *greatest fixpoint operator* to express infinite behavioural patterns. For each action $a \in Act$ the operators $[a]$ and $\langle a \rangle$ mean 'after every' and 'after some' a-transition respectively. *Atomic sentences* $A \subseteq Act \cup \{\varepsilon\}$ restrict the set of possibles moves and the divergence potential to members of A. Altogether $\mathcal{L}$ is given by:

$$F \ ::= \ X \mid A \mid F \wedge F \mid F \vee F \mid [a]F \mid \langle a \rangle F \mid \nu X.F$$

where $A \subseteq Act \cup \{\varepsilon\}$, $a \in Act$ and X ranges over a set of variables. It is convenient to define the n-th approximation of a fixpoint formula as a derived operator:

$$(\nu X\, F)^n \ where\ (\nu X\, F)^0 =_{df} tt\ and\ (\nu X\, F)^{n+1} =_{df} F[X := (\nu X\, F)^n]$$

where $[X := f]$ means syntactic substitution of the free occurrences of X by f and tt abbreviates the atomic sentence $Act \cup \{\varepsilon\}$, which is satisfied by every process of $\mathcal{P}_c$ (see below). Furthermore we abbreviate the set of all closed formulae in $\mathcal{L}$ by $\mathcal{L}_c$.

We directly interpret the *closed* formulae of the modal language $\mathcal{L}$ on transition systems T by inductively defining the satisfaction relation $\models$ between states of T and formulae of $\mathcal{L}_c$.

$$\begin{array}{lll}
p \models A & \text{iff} & \{\mu : p \Uparrow \mu \vee \exists\, q \in \mathbf{P}.\, p \xrightarrow{\mu} q\} \subseteq A \\
p \models F \wedge G & \text{iff} & p \models F\ and\ p \models G \\
p \models F \vee G & \text{iff} & p \models F\ or\ p \models G \\
p \models [a]F & \text{iff} & p \Downarrow a\ and\ \forall q \in \mathbf{P}.\ if\ p \xrightarrow{a} q\ then\ q \models F \\
p \models \langle a \rangle F & \text{iff} & \exists\, q \in \mathbf{P}.\, p \xrightarrow{a} q\ and\ q \models F \\
p \models \nu X.F & \text{iff} & \forall n.\, p \models (\nu X\, F)^n
\end{array}$$

Note, the convergence condition on $[a]$. So, there is an asymmetry between $[a]$ and $\langle a \rangle$ in that they are not duals. This is the basis for an intuitionistic understanding of modal logics as in [17, 19]. Indeed, the asymmetry in the intuitionistic interpretation of the logical language is necessary to cover the asymmetry of the preorder of our process language. A classical interpretation, for example, would already fail to deal with the existence of a smallest element, which exists in form of the totally divergent process in our process language. Intuitionistic interpretation however, delivers the following characterization theorem, which extends the results of [15, 19]. This theorem is a consequence of the Main Theorem 4.12.

Theorem 3.1 $\forall p, q \in \mathbf{P}.\ p \preceq q$ *iff* $(\forall F \in \mathcal{L}_c.\ p \models F$ *implies* $q \models F)$

So, $p \preceq q$ means that q has more properties (expressible in $\mathcal{L}_c$) than p. Classical negation would exclude the proper inclusion of the set of properties of one process in the set of properties of another process. That is the reason for classical interpretations to be suitable for characterizing equivalences rather than preorders.

4 Characteristic Formulae

Unlike [15, 19], where logics do not contain the important fixpoint operator, there are *characteristic formulae* for $\preceq$ in $\mathcal{L}_c$, i.e. for each process p there exists a formula which is satisfied by a process q, iff $p \preceq q$. Characteristic formulae have been introduced by Graf and Sifakis in [9], where it was shown how to derive a logical specification for finite processes up to observational conguence (i.e. characteristic formulae for $\approx^c$) in the modal mu-calculus [11]. We are now going to show, how to derive characteristic formulae $\| p \|$ for processes $p \in \mathbf{P}$ in our setting. This can be done in two steps:

1. The transformation of p into a closed algebraic term with *fixpoint operators.*
2. The successive construction of the formula $\| p \|$ using a semantic functional, the *characteristic functional.*

4.1 The Transformation

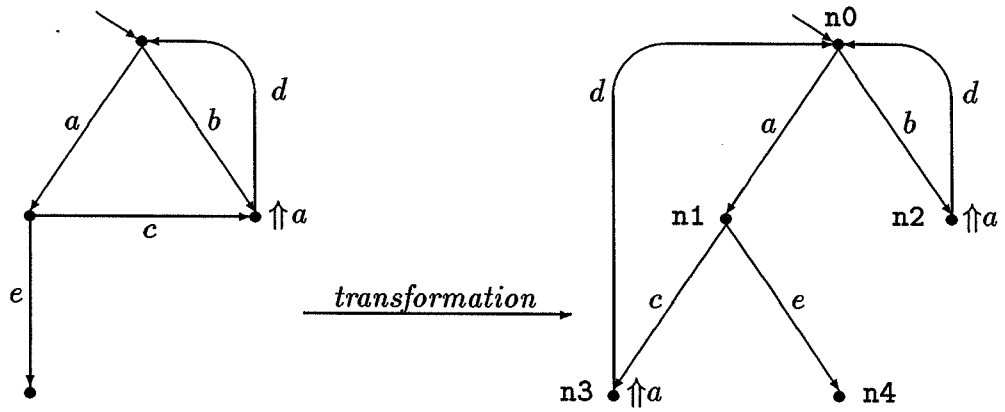
The characteristic functional needs a hierarchical representation of its argument process p, which it explores in depth first manner to finally arrive at a formula of $\mathcal{L}_c$. We will construct the required representation for p using the same idea, which is generally used to transform a mutually recursive equation system into an equivalent hierarchy of nested recursions [4, 13]. For our context, such hierarchies can be expressed in the following process language $\mathcal{P}$:

$$p \ ::= \ X \mid fix(X,p) \mid \sum\{a_i.p_i : i \in \mathcal{I}\} + \sum\{\Omega_\mu : \mu \in \mathcal{A}\}$$

where X denotes a process variable, which indicates recursive 'calls', $fix(X,p)$ represents 'declarations' of processes, which are 'called' recursively and the fourth alternative expresses the branching structure ($\sum\{a_i.p_i : i \in \mathcal{I}\}$, where $a_i \in Act$ and $\mathcal{I}$ is a finite set of indices) and the divergence potential ($\sum\{\Omega_\mu : \mu \in \mathcal{A}\}$, where $\mathcal{A} \subseteq Act \cup \{\varepsilon\}$) of a process p. Note that $\sum\{\,\} + \sum\{\,\}$ represents the process $\mathbf{0}$ (cf. [14, 16]), which can neither diverge nor perform any action.

Two subsets of $\mathcal{P}$ are of particular interest. First, the subset $\mathcal{P}_g$ of all expressions with *strongly guarded* recursion only, i.e. admissible fixpoint expressions are of the form $fix\,(X, \sum\{a_i.p_i : i \in \mathcal{I}\} + \sum\{\Omega_\mu : \mu \in \mathcal{A}\})$. Second, the subset $\mathcal{P}_c$ of all closed expressions of $\mathcal{P}_g$. Indeed, given a process p of a transition system T, our transformation always ends up in $\mathcal{P}_c$.

The transformation proceeds in two steps. The first step transforms the original process into a *regular tree* (i.e. a tree with back arcs [4, 13]) whose nodes are labelled uniquely. Instead of giving the formal definition, we illustrate this transformation by means of an example:



The first transformation step does not affect the *derivation tree*[5] of its argument process. Thus it maintains the preorder and the satisfaction relation:

Lemma 4.1 *Let $p, q \in \mathbf{P}$ with $\tilde{p}$ and $\tilde{q}$ as their transformations into regular trees respectively, and $F \in \mathcal{L}_c$. Then we have:*

1. $p \preceq q$ *iff* $\tilde{p} \preceq \tilde{q}$
2. $p \models F$ *iff* $\tilde{p} \models F$

[5]The derivation tree of a transition system is the (usually infinite) tree, which is generated by unrolling loops successively (cf. [14, 16]).

Note that $\tilde{p}$ and $\tilde{q}$ essentially are elements of $\mathbf{P}$. Thus we do not have to modify $\preceq$ and $\models$ when dealing with processes resulting from the first transformation step.

The second step translates regular trees into the process language $\mathcal{P}_g$. It proceeds in *depth first* manner along the tree structure of the regular trees. Thus we can assume that all the sons of a node which is going to be processed are already translated. Only back arcs reach unprocessed nodes. This is where variables come into play. We describe the second translation step algorithmically in two steps. It assigns process expressions <n> to each node n of the regular tree under consideration. The actual result of this translation process is the expression assigned to the start node <n0>.

1. Initialization: $\texttt{<ni>} =_{df} X_i \quad (X_i \neq X_j \text{ for } i \neq j)$
2. Construction of the process expression for inner nodes in depth first order:

$$\texttt{<ni>} =_{df} fix(X_i, \sum\{a.\texttt{<m>} : \texttt{ni} \xrightarrow{a} \texttt{<m>}\} + \sum\{\Omega_\mu : \texttt{ni} \Uparrow \mu\})$$

Note, this translation process necessarily produces a closed formula. In fact, introducing the notion $tr : \mathbf{P} \to \mathcal{P}$ for the whole two step translation process, we obtain:

Lemma 4.2 $\forall p \in \mathbf{P}.\ tr(p) \in \mathcal{P}_c$

Expressions of $\mathcal{P}_c$ are interpreted as one would expect, i.e. by means of the following two derivation rules:

$$\text{Gen}\ \frac{-}{\sum\{a_i.p_i : i \in \mathcal{I}\} + \sum\{\Omega_\mu : \mu \in \mathcal{A}\} \xrightarrow{a_i} p_i}$$

$$\text{Fix}\ \frac{p[X := fix(X,p)] \xrightarrow{a} q}{fix(X,p) \xrightarrow{a} q}$$

Extending the notions of preorder, $\preceq$, and satisfaction, $\models$, to expressions of $\mathcal{P}_c$ by means of these derivation rules and interpreting a summand Ω_μ as μ-divergence ($\Uparrow \mu$), Lemma 4.1 and Lemma 4.2 deliver:

Lemma 4.3 *Let $p, q \in \mathbf{P}$ and $F \in \mathcal{L}_c$. Then we have:*

1. $p \preceq q$ *iff* $tr(p) \preceq tr(q)$
2. $p \models F$ *iff* $tr(p) \models F$

4.2 Extensions, Approximations and Fixpoint Properties

The main part of the proof of the Main Theorem 4.12 consists of the verification of the Main Lemma 4.11, which is a reformulation of the Main Theorem 4.12 for processes in $\mathcal{P}_g$. The induction step of the proof of this lemma essentially is a case analysis according to the structure of expressions of $\mathcal{P}_g$ and $\mathcal{L}$, including *open process expressions* and *open formulae*. We therefore need to extend our notions of preorder, $\preceq$, and satisfaction, $\models$, using $[(X_1,\cdots,X_n):=(r_1,\cdots,r_n)]$ for simultaneous syntactic substitution of the (free occurrences of the) X_i by the corresponding r_i:

Definition 4.4 *Let $p,q \in \mathcal{P}_g$, $F \in \mathcal{L}$ and $\bar{X} = (X_1,\cdots,X_n)$ be a vector containing all the variables as components which occur free in p, q or F. Then we define the extended preorder, $\preceq_e$, and the extended satisfaction relation, $\models_e$, by:*

1. $p \preceq_e q$ *iff* $\forall \bar{r} \in \mathcal{P}_c^n.\ p[\bar{X}:=\bar{r}] \preceq q[\bar{X}:=\bar{r}]$
2. $p \models_e F$ *iff* $\forall \bar{r} \in \mathcal{P}_c^n\, \forall \bar{G} \in \mathcal{L}_c^n.\ \bar{r} \models \bar{G}$ *implies* $p[\bar{X}:=\bar{r}] \models F[\bar{X}:=\bar{G}]$ *where* $\bar{r} \models \bar{G}$ *is understood componentwise.*

Furthermore, we consider *finite approximations* of $\preceq_e$ and $\models_e$. The proof of the Main Lemma 4.11 proceeds by induction on the 'depth' of these approximations.

Definition 4.5 *Let $p,q \in \mathcal{P}_g$ and $\bar{X} = (X_1,\cdots,X_n)$ be a vector containing all the variables as components which occur free in p or q. Then the extended k-limited preorder, $\preceq_{e,k}$, is defined inductively on the size of k:*

1. $p \preceq_{e,0} q$ *holds always*
2. *If* $k \geq 1$ *then* $p \preceq_{e,k} q$ *holds, iff*
   (a) $\forall \bar{r} \in \mathcal{P}_c^n\, \forall a \in \mathrm{Act}\, \forall p' \in \mathcal{P}_c.$ *if* $p[\bar{X}:=\bar{r}] \xrightarrow{a} p'$ *then* $\exists q' \in \mathcal{P}_c.\ q[\bar{X}:=\bar{r}] \xrightarrow{a} q'$ *and* $p' \preceq_{e,k-1} q'$
   (b) $\forall a \in \mathrm{Act}.$ *if* $p \Downarrow a$ *then* ($q \Downarrow a$ *and* $\forall \bar{r} \in \mathcal{P}_c^n\, \forall q' \in \mathcal{P}_c.$ *if* $q[\bar{X}:=\bar{r}] \xrightarrow{a} q'$ *then* $\exists p' \in \mathcal{P}_c.\ p[\bar{X}:=\bar{r}] \xrightarrow{a} p'$ *and* $p' \preceq_{e,k-1} q'$)

Extended k-limited preorders can be used to characterize the notion of extended preorder:

Theorem 4.6 $\forall p,q \in \mathcal{P}_g.\ p \preceq_e q$ *iff* $\forall k \geq 0.\ p \preceq_{e,k} q$

Whereas the implication "$\Rightarrow$" is straightforward, the converse depends on the considered processes having only a finite number of states. In fact, *image finiteness* of the transition relations would be sufficient.

A k-limited version of the satisfaction relation can be defined in an analogous way:

Definition 4.7 *Let $p \in \mathcal{P}_g$ and $\bar{X} = (X_1, \cdots, X_n)$ be a vector containing all the variables as components which occur free in p. Then the extended k-limited satisfaction relation, $\models_{e,k}$, is defined inductively on the size of k:*

1. *$p \models_{e,0} F$ holds for every $p \in \mathcal{P}_g$ and $F \in \mathcal{L}$*
2. *If $k \geq 1$ then $\models_{e,k}$ is given by:*

$$
\begin{array}{lll}
p \models_{e,k} X & \textit{iff} & p \models_e X \\
p \models_{e,k} A & \textit{iff} & p \models_e A \\
p \models_{e,k} F \wedge G & \textit{iff} & p \models_{e,k} F \textit{ and } p \models_{e,k} G \\
p \models_{e,k} F \vee G & \textit{iff} & p \models_{e,k} F \textit{ or } p \models_{e,k} G \\
p \models_{e,k} [a]F & \textit{iff} & p \Downarrow a \textit{ and } (\forall \bar{r} \in \mathcal{P}_c^n \, \forall q \in \mathcal{P}_c. \\
 & & \textit{if } p[\bar{X} := \bar{r}] \xrightarrow{a} q \textit{ then } q \models_{e,k-1} F) \\
p \models_{e,k} \langle a \rangle F & \textit{iff} & \forall \bar{r} \in \mathcal{P}_c^n \, \exists q \in \mathcal{P}_c. \\
 & & p[\bar{X} := \bar{r}] \xrightarrow{a} q \textit{ and } q \models_{e,k-1} F \\
p \models_{e,k} \nu X.F & \textit{iff} & \forall n.\ p \models_{e,k} (\nu X\, F)^n
\end{array}
$$

leading to a similar characterization result:

Theorem 4.8 $\forall p \in \mathcal{P}_g \, \forall F \in \mathcal{L}.\ \ p \models_e F \ \ \textit{iff} \ \ \forall k \geq 0.\ p \models_{e,k} F$

Finally, we present three properties relating fixpoint processes and fixpoint formulae by means of $\models_{e,k}$ and $\preceq_{e,k}$:

Theorem 4.9 (Fixpoint Properties)

Let $p \in \mathcal{P}_g$ and $F \in \mathcal{L}$. Then we have:

1. $p \models_{e,k} \nu X.F \ \ \textit{iff} \ \ p \models_{e,k} F[X := \nu X.F]$
2. $fix(X,p) \models_{e,k} F \ \ \textit{iff} \ \ p[X := fix(X,p)] \models_{e,k} F$
3. $fix(X,p) \preceq_{e,k} p[X := fix(X,p)] \preceq_{e,k} fix(X,p)$

Whereas the proof of the first property is trivial, the remaining two properties depend on the fact that $fix(X,p)$ and $p[X := fix(X,p)]$ possess identical derivation trees.

4.3 The Characteristic Functional

The *characteristic functional* $|\,.\,| : \mathcal{P}_g \longrightarrow \mathcal{L}$ is recursively defined on the structure of its argument process:

$$|\,X\,| =_{df} X$$

$$\begin{aligned} |\,\Sigma\{a_i.p_i : i\in\mathcal{I}\}+\Sigma\{\Omega_\mu : \mu\in\mathcal{A}\}\,| =_{df}\ & \wedge\{\langle a_i\rangle\,|p_i|\ :\ i\in\mathcal{I}\}\ \wedge \\ & \wedge\{\,[a_i]\,\vee\{\,|p_j|\ :\ j\in\mathcal{I}\wedge a_j=a_i\}\ : \\ & \quad i\in\mathcal{I}\wedge a_i\notin\mathcal{A}\} \\ & \wedge\ \{a_i : i\in\mathcal{I}\}\cup\mathcal{A} \end{aligned}$$

$$|\,fix(X,q)\,| =_{df} \nu X.\,|\,q\,|$$

First, we obtain the compositionality of the characteristic functional.

Theorem 4.10 (Compositionality Theorem)

$\forall p,q\in\mathcal{P}_g.\ \ |\,q[X:=p]\,| \ =\ |\,q\,|[X:=|\,p\,|]$

Proof: We proceed by induction on the structure of q. The case where q is a variable is trivial. Thus let us next assume that $q=\Sigma\{a_i.q_i : i\in\mathcal{I}\}+\Sigma\{\Omega_\mu : \mu\in\mathcal{A}\}$ for some $a_i\in Act$ and $q_i\in\mathcal{P}_g$. Then this case is a consequence of the following sequence of equalities:

$$\begin{aligned} & |\,(\,\Sigma\{a_i.q_i : i\in\mathcal{I}\}+\Sigma\{\Omega_\mu : \mu\in\mathcal{A}\}\,)\,[X:=p]\,| \\ = \ & |\,(\,\Sigma\{a_i.q_i[X:=p] : i\in\mathcal{I}\}+\Sigma\{\Omega_\mu : \mu\in\mathcal{A}\}\,)\,| \\ = \ & \wedge\{\langle a_i\rangle\,|q_i[X:=p]|\ :\ i\in\mathcal{I}\}\ \wedge \\ & \wedge\{\,[a_i]\,\vee\{\,|q_j[X:=p]|\ :\ j\in\mathcal{I}\wedge a_j=a_i\}\ : \\ & \quad i\in\mathcal{I}\wedge a_i\notin\mathcal{A}\} \\ & \wedge\ \{a_i : i\in\mathcal{I}\}\cup\mathcal{A} \\ (\text{induction hypothesis}) = \ & \wedge\{\langle a_i\rangle\,|q_i|\,[X:=|p|]\ :\ i\in\mathcal{I}\}\ \wedge \\ & \wedge\{\,[a_i]\,\vee\{\,|q_j|\,[X:=|p|]\ :\ j\in\mathcal{I}\wedge a_j=a_i\}\ : \\ & \quad i\in\mathcal{I}\wedge a_i\notin\mathcal{A}\} \\ & \wedge\ \{a_i : i\in\mathcal{I}\}\cup\mathcal{A} \\ = \ & |(\,\Sigma\{a_i.q_i : i\in\mathcal{I}\}+\Sigma\{\Omega_\mu : \mu\in\mathcal{A}\}\,)|\ [X:=|p|] \end{aligned}$$

Finally let us assume that q is of the form $q=fix(Y,q')$ with $X\neq Y$. This allows the following completion of the proof:

$$
\begin{array}{lcl}
|\, fix(Y,q')\,|\,[X:=\,|p|\,] & = & (\,\nu Y.\,|q'|\,)\,[X:=\,|p|\,] \\
 & = & \nu Y.\,(\,|q'|\,[X:=\,|p|\,]\,) \\
\text{(induction hypothesis)} & = & \nu Y.\,|q'\,[X:=p]\,| \\
 & = & |\, fix(Y,q')\,[X:=p]\,|
\end{array}
$$

□

The central property of the characteristic functional, however, is that it characterizes process expressions up to $\preceq_e$:

Lemma 4.11 (Main Lemma)

$\forall p,q \in \mathcal{P}_g.\ p \preceq_e q \ \ \textit{iff} \ \ q \models_e |\, p\,|$

Proof: According to Theorem 4.6 and Theorem 4.8 it is obviously enough to prove:

$(*)\ \ \forall k \geq 0\ \forall p,q \in \mathcal{P}_g.\ \ p \preceq_{e,k} q \ \text{ iff } \ q \models_{e,k} |\, p\,|$

The advantage of $(*)$ is that it can be proven by induction on k. The case $k = 0$ is trivial. Thus let $k \geq 1$. Then we distinguish four cases according to the structure of p. The case where p is a variable is trivial. Thus let us next assume that $p = fix(X,p')$ for some $p' \in \mathcal{P}_g$. Then the Fixpoint Properties 4.9 (1 & 3) imply:

- $q \models_{e,k} |\, p\,| \ \text{ iff } \ q \models_{e,k} |\, p'\,|\,[X := \nu X.\,|\,p'\,|\,]$
- $p \preceq_{ek} p'\,[X := fix(X,p')] \preceq_{ek} p$

respectively. In the light of the Compositionality Theorem 4.10 it is therefore sufficient to prove the following case:

$$\forall p,q \in \mathcal{P}_g.\ \ p'\,[X := fix(X,p')] \preceq_{e,k} q \ \text{ iff } \ q \models_{e,k} |\, p'\,[X := fix(X,p')]\,|$$

Thus, according to the strong guardedness of p, it only remains to deal with the fourth case, where p is of the form $p = \sum\{a_i.p_i : i \in \mathcal{I}\} + \sum\{\Omega_\mu : \mu \in \mathcal{A}\}$ and therefore, where:

$$
\begin{array}{rcl}
|\, p\,| & =_{df} & \bigwedge\{\langle a_i\rangle\,|p_i| \,:\, i \in \mathcal{I}\} \ \wedge \\
 & & \bigwedge\{\,[a_i]\,\bigvee\{\,|p_j| \,:\, j \in \mathcal{I} \wedge a_j = a_i\} : \\
 & & \quad i \in \mathcal{I} \wedge a_i \notin \mathcal{A}\} \\
 & & \wedge\ \{a_i : i \in \mathcal{I}\} \cup \mathcal{A}
\end{array}
$$

On the other hand, investigating both sides of the equivalence $(*)$ under these circumstances, we observe on either side that q cannot be a variable. Thus let us assume that $q = fix(X, q')$ for some $q' \in \mathcal{P}_g$. Then the Fixpoint Properties 4.9 (2 & 3) deliver:

- $q \models_{e,k} |\, p \,|$ iff $q'[X := fix(X, q')] \models_{e,k} |\, p \,|$
- $q \preceq_{ek} q'[X := fix(X, q')] \preceq_{ek} q$

respectively. And again, because of the Compositionality Theorem 4.10 this case reduces to the case where q is of the form $q = \sum\{b_j.q_j : j \in \mathcal{I}'\} + \sum\{\Omega_\nu : \nu \in \mathcal{A}'\}$. Thus $p \preceq_{e,k} q$ is equivalent to:

(i) $\forall i \in \mathcal{I}\ \exists j \in \mathcal{I}'.\ a_i = b_j$ and $p_i \preceq_{e,k-1} q_j$ and

(ii) $\mathcal{A}' \subseteq \mathcal{A}$ and $(\forall j \in \mathcal{I}'.\ b_j \in \mathcal{A}$ or $(\exists i \in \mathcal{I}.\ a_i = b_j$ and $p_i \preceq_{e,k-1} q_j))$

Together with the induction hypothesis, we therefore obtain the following equivalent to $p \preceq_{e,k} q$:

(1) (i) $\forall i \in \mathcal{I}\ \exists j \in \mathcal{I}'.\ a_i = b_j$ and $q_j \models_{e,k-1} |p_i|$ and

(ii) $\mathcal{A}' \subseteq \mathcal{A}$ and $(\forall j \in \mathcal{I}'.\ b_j \in \mathcal{A}$ or $(\exists i \in \mathcal{I}.\ a_i = b_j$ and $q_j \models_{e,k-1} |p_i|))$

Thus it remains to show that (1) is equivalent to $q \models_{e,k} |\, p \,|$.

To prove "$\Rightarrow$", let us assume (1), and let F be a syntactically minimal conjunct of $|\, p \,|$. Then we are going to show $q \models_{e,k} F$, distinguishing three cases according to the structure of F.

Due to 1(ii) we know that $\mathcal{A}' \subseteq \mathcal{A}$ and $\{b_j : j \in \mathcal{I}'\} \subseteq \{a_i : i \in \mathcal{I}\} \cup \mathcal{A}$. This shows $q \models_{e,k} F$ in case of $F = \{a_i : i \in \mathcal{I}\} \cup \mathcal{A}$.

Let now $F = \langle a_i \rangle |p_i|$ for some $i \in \mathcal{I}$. Then 1(i) and the construction of $|\, p \,|$ deliver the existence of a $j \in \mathcal{I}'$ with $a_i = b_j$ and $q_j \models_{e,k-1} |\, p_i \,|$, and therefore $q \models_{e,k} F$ as desired.

Finally, let $F = [a_i] \bigvee \{|p_{i'}| : i' \in \mathcal{I} \wedge a_{i'} = a_i\}$ for some $i \in \mathcal{I}$. Then the construction of $|p|$ gives us $a_i \notin \mathcal{A}$ and therefore $a_i \notin \mathcal{A}'$ because of 1(ii). Thus it remains to show:

$$\forall j \in \mathcal{I}'.\ a_i = b_j \text{ implies } q_j \models_{e,k-1} \bigvee \{|p'_i| : i' \in \mathcal{I} \wedge a_{i'} = a_i\}$$

Whence let $j \in \mathcal{I}'$ with $a_i = b_j$. Then $b_j \notin \mathcal{A}$ and 1(ii) guarantee the existence of an $i' \in \mathcal{I}$ satisfying $q_j \models_{e,k-1} |p_{i'}|$, which completes the proof of the first implication.

To show the converse, "$\Leftarrow$", let $q \models_{e,k} |\, p \,|$. Then we must verify the two properties formulated under (1). Thus let $i \in \mathcal{I}$. Then the definition of $|\,.\,|$ delivers

a conjunct $\langle a_i \rangle |p_i|$ of $|p|$ and therefore the existence of a q' with $q \xrightarrow{a_i} q'$ and $q' \models_{e,k-1} |p_i|$. According to the pattern of q this yields 1(i).

Let now $b \in \mathcal{A}'$. Then the assumption of $b \notin \mathcal{A}$ would lead to $b \in \{a_i : i \in \mathcal{I}\}$ and therefore to $q \models_{e,k} [b] \bigvee \{|p_i| : i \in \mathcal{I} \wedge b = a'_i\}$ in contradiction to $b \in \mathcal{A}'$. Thus it suffices to finally consider a $j \in \mathcal{I}'$ with $b_j \notin \mathcal{A}$, and consequently with $b_j \in \{a_i : i \in \mathcal{I}\} \setminus \mathcal{A}$. This yields $q \models_{e,k} [b_j] \bigvee \{|p_i| : i \in \mathcal{I} \wedge b_j = a_i\}$ and therefore the existence of an $i \in \mathcal{I}$ with $b_j = a_i$ and $q_j \models_{e,k-1} |p_i|$. □

After having analyzed the two steps of the procedure for generating characteristic formulae for processes $p \in \mathbf{P}$, we are now able to collect our main result. Let therefore $\| . \| =_{df} | . | \circ tr$. Then we obtain as an immediate consequence of the Main Lemma 4.11 and Lemma 4.3:

Theorem 4.12 (Main Theorem)

$$\forall p, q \in \mathbf{P}.\ p \preceq q \quad \textit{iff} \quad q \models \| p \|$$

5 Application to CCS

In CCS, bisimulation–like relations are used to compare the computational power of processes. Depending on the needed sensitivity (deadlock, convergence, etc.) and the mathematical covenience (axiomatizability, computability and provability) one can choose between $\sqsubseteq$ (strong preorder), $\lesssim$ (weak preorder), $\sim$ (strong equivalence), $\approx$ (observational or weak equivalence) and $\approx^c$ (observational congruence). — The definitions of the strong preorder, the weak preorder and the equivalences can be found in [19], [21][6] and [14, 16] respectively. — We are now going to show how to obtain characteristic formulae for each of these relations.

The strong preorder, $\sqsubseteq$, is just $\preceq$ in the context of a CCS–like transitions system. Thus the strong preorder case is covered by means of the Main Theorem 4.12. For the other four relations we define four process transformations: W (for 'weakening' the transition relation by collapsing τ–actions, which are regarded as *internal* or *unobservable*), C (to obtain convergence), E (to extend the corresponding transition system in a way that excludes the start state as the destination of a transition) and S (for 'weakening', while maintaining the sensitivity for τ-moves of the start state). As before we represent processes as rooted transition systems (i.e. as pairs (T, p_0), where p_0 is the start state):

- $W((\mathbf{P}, \rightarrow, Act, \Uparrow), p_0) =_{df} ((\mathbf{P}, \Rightarrow, (Act \setminus \{\tau\}) \cup \{\epsilon\}, \Uparrow), p_0)$, where

  1. $\forall p, q \in \mathbf{P}\ \forall a \in Act \setminus \{\tau\}.\ p \stackrel{a}{\Longrightarrow} q$ iff $p \xrightarrow{\tau}{}^* \xrightarrow{a} \xrightarrow{\tau}{}^* q$ and

[6] Walker denotes the weak preorder by $\sqsubseteq$.

2. $\forall p,q \in \mathbf{P}.\ p \stackrel{\epsilon}{\Longrightarrow} q$ iff $p \stackrel{\tau}{\longrightarrow}^* q$

- C only changes the divergence predicate $\Uparrow$ to the predicate $\uparrow$ that is constantly false.
- $E((\mathbf{P}, \rightarrow, Act, \Uparrow), p_0) =_{df} ((\mathbf{P} \cup \{\bar{p_0}\}, \rightarrow', Act, \Uparrow'), \bar{p_0})$, where $\rightarrow'$ and $\Uparrow'$ are the smallest extensions of $\rightarrow$ and $\Uparrow$ respectively, such that $\bar{p_0}$ and p_0 possess identical branching structures and divergence potentials (i.e. $\bar{p_0}$ is a copy of p_0).
- $S((\mathbf{P}, \rightarrow, Act, \Uparrow), p_0) =_{df} ((\mathbf{P}, \Rightarrow, Act \cup \{\epsilon\}, \Uparrow), p_0)$, where

  1. $\forall p,q \in \mathbf{P}\ \forall a \in Act.\ p \stackrel{a}{\Longrightarrow} q$ iff $p \stackrel{\tau}{\longrightarrow}^* \stackrel{a}{\longrightarrow} \stackrel{\tau}{\longrightarrow}^* q$
  2. $\forall p,q \in \mathbf{P}.\ p \stackrel{\epsilon}{\Longrightarrow} q$ iff $p \stackrel{\tau}{\longrightarrow}^* q$ and

  In particular, the transition relation "$\Rightarrow$" resulting from this transformation records weak τ–moves of the start state, i.e.: $p_0 \stackrel{\tau}{\Longrightarrow}$ iff $p_0 \stackrel{\tau}{\longrightarrow}^+$.

Note, the distinction between ε and ϵ. This distinction originates from the fact that both, the weakening transformation and the global divergence predicate require their private actions. However, it is possible and often convenient to identify ε and ϵ, which actually both represent unobservale actions. One only must change the satisfaction relation $\models$ for atomic sentences A as follows:

$$p \models A \text{ iff } \{\mu : p \Uparrow \mu \vee (\mu \neq \varepsilon \wedge \exists q \in \mathbf{P}.\, p \stackrel{\mu}{\longrightarrow} q)\} \subseteq A$$

According to the Main Theorem 4.12 characteristic formulae for the remaining four relations can now be obtained by means of the following properties of the transformations just defined:

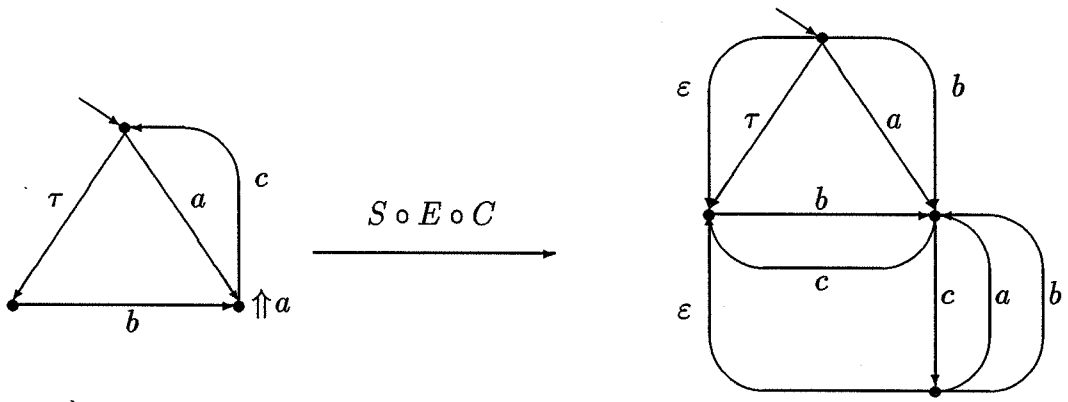
Theorem 5.1 *Let $p,q \in \mathbf{P}$. Then we have:*

1. $p \precsim q$ *iff* $W(q) \sqsubseteq W(p)$
2. $p \sim q$ *iff* $C(q) \sqsubseteq C(p)$
3. $p \approx q$ *iff* $W(C(q)) \sqsubseteq W(C(p))$
4. $p \approx^c q$ *iff* $S(E(C(q))) \sqsubseteq S(E(C(p)))$

W is the standard transformation for changing from the strong to the weak version of equivalence and preorder; and obviously, $\sqsubseteq$(strong preorder) coincides with $\sim$(strong equivalence) in the absence of divergence. This yields the first three properties of the transformations.

For observational congruence, $\approx^c$, the 'weakening' transformation must be refined to additionally record initial τ-moves. This refinement is realized in two

steps. First, a new start node is created, which exactly 'behaves' like the original start node, but which is not reachable from inside the transition system (see E). This guarantees that a special treatment of the (new) start node only affects initial moves. Second, the process is transformed into its weak version, in a way, which records the possible τ-moves of the (new) start state (see S). The following figure illustrates this transformation:



For clarity, the right hand side diagram does not display ε–edges that result from the reflexive closure of $\xrightarrow{\tau}$.

6 Conclusion

The results presented in this paper do not only demonstrate the expressive power of an intuitionistic interpreted modal mu–calculus, they also build the theoretical background for a uniform method for the *automatic verification* of bisimulation–like relations between processes by means of *model checking*: transform the underlying transition system w.r.t. the chosen process relation as described in section 5 [7], construct the characteristic formula for the left hand side process and check whether the right hand side process satisfies the characteristic formula constructed in the second step. Whereas the first two steps of this method can be implemented directly along the lines described in this paper, the third step can be realized as a modification of existing model checking algorithms like the ones in [1, 3, 8, 12, 20].

[7]This is the only part which depends on the process relation under consideration.

7 Acknowledgements

I would like to record my appreciation to Colin Stirling who introduced me to the problem studied in this paper. His critical remarks were of great help. Furthermore I would like to thank Rance Cleaveland, Michael Mendler, Robin Milner, Ernst–Rüdiger Olderog, Joachim Parrow and Bent Thomsen for their constructive comments. I have been supported by a grant from the Science and Engineering Research Council.

References

[1] E. Clarke, E. A. Emerson and A.P. Sistla. *Automatic Verification of Finite State Concurrent Systems using Temporal Logic Specifications: A Practical Approach*, ACM 1983

[2] E. Clarke, O. Grumberg and M.C. Browne. *Reasoning About Networks With Many Identical Finite–State Processes*, Carnegie Mellon University, Pittsburg, October 1986

[3] R. Cleaveland. *Tableau-Based Model Checking in the Propositional Mu-Calculus*, University of Sussex, Brighton, Technical Report 2-89, 1989

[4] G. Cousineau and M. Nivat. *On Rational Expressions Representing Infinite Rational Trees: Application to the Structure of Flow Charts*, 8th MFCS, LNCS 74, pp. 567-580, 1979

[5] R. Cleaveland, J. G. Parrow and B. Steffen. *The Concurrency Workbench: Operating Instructions*, University of Edinburgh, Laboratory for Foundations of Computer Science, Technical Notes 10, September 1988

[6] R. Cleaveland, J. G. Parrow and B. Steffen. *The Concurrency Workbench*, Accepted for the Workshop on Automatic Verification Methods for Finite State Systems, Grenoble, France, 1989

[7] R. Cleaveland, J. G. Parrow and B. Steffen. *A Semantics based Verification Tool for Finite State Systems*, to appear in the proceedings of the Ninth International Symposium on Protocol Specification, Testing, and Verification; North Holland, 1989

[8] E. A. Emerson and C.-L. Lei. *Efficient Model Checking in Fragments of the Propositional Mu-Calculus*, LICS, Cambridge, Mass., IEEE, 1986

[9] S. Graf and J. Sifakis. *A Modal Characterization of Observational Congruence on Finite Terms of CCS*, Information and Control, pp. 125-145, Vol 68, 1986

[10] S. Holmström. *Hennessy-Milner Logic with Recursion as a Specification Language, and a Refinement Calculus based on it*, Report 44 Programming Methodology Group, University of Göteborg, 1988

[11] D. Kozen. *Results on the Propositional Mu-Calculus*, TCS 27, pp. 333-354, North Holland, 1983

[12] K. Larsen. *Proof Systems for Hennessy–Milner Logic with Recursion*, in Proceedings CAAP 1988

[13] A.R. Meyer and K.A. Winklmann. *On the Expressive Power of Dynamic Logics*, 11th STOC, ACM, pp. 167-175, 1979

[14] R. Milner. *A Calculus for Communicating Systems*, LNCS 92

[15] R. Milner. *A Modal Characterization of Observable Machine-Behaviour*, LNCS 112, pp. 25-34, Berlin 1981

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

[17] G. Plotkin and C. Stirling. *A Framework for Intuitionistic Modal Logics*, Theoretical Aspects of Reasoning about Knowledge, Monterey, 1986

[18] B. Steffen. *Characteristic Formulae*, in the Proceedings ICALP 1989

[19] C. Stirling. *Modal Logics for Communicating Systems*, TCS 49, pp. 311-347, 1987

[20] C. Stirling and D. J. Walker. *Local Model Checking in the Modal Mu-Calculus*, in Proceedings CAAP 1989

[21] D.J. Walker. *Bisimulation and Divergence in CCS*, in Proceedings LICS 1988