Process Algebra Approach to Reasoning About Concurrent Actions

Yuan Feng and Ming-Sheng Ying

State Key Laboratory of Intelligent Technology and Systems, Department of Computer Science and Technology
Tsinghua University, Beijing 100084, P.R. China
E-mail: fengy99g@mails.tsinghua.edu.cn

Received February 27, 2001; revised July 8, 2002.

Abstract A reasonable transition rule is proposed for synchronized actions and some equational properties of bisimilarity and weak bisimilarity in the process algebra for reasoning about concurrent actions are presented.

Keywords process, global store, constraint set, bisimulation

1 Introduction

Reasoning about actions is a very active research field in artificial intelligence. The major thesis is specifying and reasoning about dynamic systems which may evolve in performing actions. Various formal methods for this purpose have been proposed, e.g., see [1, 2]. The typical methodology of specifying dynamic systems is to introduce a set of facts whose values change as the system evolves and the effects of actions on these facts. This allows us to describe state transitions resulting from executing certain actions, see [1] for details. The typical reasoning problem in the field is logical implication. More explicitly, we introduce a certain logic which is suitable to describe dynamic systems and then with this logic we deduce some properties of states of the system under consideration from a set of hypotheses about the system.

Chen and De Giacomo [3] presented a new approach to reasoning about actions by employing some modelling tools from process algebras [4]. They constructed an algebra of processes in which each process is accompanied by a global store characterizing certain facts about the current states. This provides us with a formal method of specifying concurrent actions and allows us to organize actions within suitable control structures such as sequential and parallel compositions and nondeterministic choice. The method of specifying systems adopted in this approach may be seen as a combination of process algebra and the typical method of specifying systems in the area of reasoning about actions, and this establishes a bridge between the area of reasoning about actions in artificial intelligence and that of concurrency theory in theoretical computer science. The reasoning problem considered by Chen and De Giacomo [3] is the model checking which is different from the typical one mentioned above. They introduced a modal logic \( M_\mu \) which is suitable to describe properties of states in a dynamic system. The model checking is then to verify whether a formula in \( M_\mu \) is true in a state of a given system. The main result in [3] is a linear reduction of model checking in \( M_\mu \) to the one in standard \( \mu \)-calculus.

As usual, a structured operational semantics of the process algebra in [3] was introduced as a transition system. The major difference between this transition system and the transitional semantics of previous process algebras is that in the former a state is a configuration consisting of a process and a global store, whereas in the latter a state is merely a process. Coping with global stores in most transitions is quite easy, but the global store in the transitions caused by performing synchronized (concurrent) actions deserves a careful treatment. In [3], the rule for synchronized actions is given as follows:

\[
\text{Syn}:(p_1, \sigma) \xrightarrow{\alpha_1}(p'_1, \sigma'_1), (p_2, \sigma) \xrightarrow{\alpha_2}(p'_2, \sigma'_2)
\]

\[
(p_1 \parallel p_2, \sigma) \xrightarrow{\alpha_1 \cup \alpha_2}(p'_1 \parallel p'_2, \sigma')
\]

where \( \sigma' \) may be chosen as any global store obtained by executing the synchronized action \( \alpha_1 \cup \alpha_2 \) of \( \alpha_1 \) and \( \alpha_2 \) in \( \sigma \), and it is completely independent of the global stores \( \sigma'_1 \) and \( \sigma'_2 \). This is obviously unreasonable. One of the aims of this paper is to overcome this objection by introducing the notion of composition of global stores and presenting a more reasonable rule for synchronized actions.

*Correspondence

This work was supported by the National Natural Science Foundation of China (Grant No.60273003) and the Key Grant Project of Chinese Ministry of Education.
A bisimulation equivalence between processes was introduced in [3], but algebraic properties of this equivalence were ignored because the main purpose was to give a reduction from model checking in $M_\mu$ to model checking in standard $\mu$-calculus. As an addendum, various algebraic properties of bisimulation in the process algebra in [3] are exploited in this paper.

The issue of equivalent descriptions is very important for the representation of and reasoning on dynamic systems. In the context of process algebras, the equivalence of the two descriptions of the same system has been well investigated and various tools have been implemented for verifying the equivalences. A question that naturally arises is when the two descriptions of a dynamic system can be considered equivalent. We adopt here the most popular notion of equivalence that has been proposed in the process algebra: bisimulation equivalence. Roughly speaking, two systems are bisimilar if during every run whenever one system can perform a certain action, the other system can perform the same matching such a move.

As we all know, a system may perform a lot of actions, many of which we do not care. In fact, sometimes we can hardly determine all actions a system can perform, so it is reasonable to focus our attention on the “constraint set” that contains all actions we concentrate on. If two processes can perform the same actions in the constraint set, we say that they are weak bisimilar, i.e., if we concentrate on the actions in the constraint set, then they are undistinguishable.

We organize this paper as follows. In Section 2, we introduce the notion of composition of global stores and use it to present a reasonable rule for synchronized actions. In this section, we also recall some concepts and notations needed in a sequence of sections from [3]. In Section 3, we study thoroughly various equational properties of bisimilarity in the process algebra for reasoning about concurrent actions. A weak bisimulation is introduced in Section 4.

2 Algebra of Processes with Global Stores and Its Transitional Semantics

In this section a slightly modified process algebra based on the one in [3] for reasoning about concurrent actions is constructed. As in [3], a state of a dynamic system is represented as a configuration, which consists of a process and a global store. Processes in our setting are similar to those discussed in the previous process algebras, which will be briefly presented in Subsection 2.3. A global store is intended to describe certain properties of the current state, and it may change as the system evolves. To describe the change of a global store, the notion of update of global store is introduced in Subsection 2.1. The evolution of a system happens in performing an action. We assume a set of atomic actions. A synchronized (concurrent) action executed by a system composed of several (concurrent) subsystems is usually treated as a certain combination of some atomic actions. In [3], a synchronized action is defined as a set of atomic action, but in this paper we need a slightly generalized concept of synchronized action. In Subsection 2.2, a synchronized action is defined as a multiset of atomic actions, and its effect on a global store is discussed.

2.1 Updates of Global Stores

Let $Prop$ be a finite set of primitive propositions. Then we write $Lit$ for the set of literals-primitive propositions and their negations-over $Prop$, i.e.,

$$Lit = Prop \cup \neg Prop = \{A, \neg A : A \in Prop\}. \quad (1)$$

By a global store we mean an interpretation over $Prop$, i.e., a mapping from $Prop$ into the set $\{tt, ff\}$ of truth values.

**Definition 2.1 ([3], Definition 2.1).** Let $\sigma$ be a global store, and let $L \subseteq Lit$ be non-contradictory, i.e., for each $A \in Prop$, $A \notin L$ or $\neg A \notin L$. Then the update $\sigma \circ L$ of $\sigma$ under $L$ is defined to be a global store and for any $A \in Prop$,

$$\begin{align*}
(\sigma \circ L)(A) = & \begin{cases} 
  tt, & \text{if } A \in L; \\
  ff, & \text{if } \neg A \in L; \\
  \sigma(A), & \text{otherwise.}
  \end{cases}
\end{align*} \quad (2)$$

Obviously, $\sigma \circ L$ is the global store most similar to $\sigma$ that satisfies all literals in $L$. The following lemma indicates that the update of a global store under the union of several sets of literals is exactly the iterated update of the global store under these sets of literals.

**Lemma 2.1.** If $L_1 \cup L_2$ is not contradictory, then $(\sigma \circ L_1) \circ L_2 = \sigma \circ (L_1 \cup L_2)$.

**Proof.** It is direct from Definition 2.1. ☐

For the case that $L_1 \cup L_2$ is contradictory, $\sigma \circ (L_1 \cup L_2)$ is not defined, but $(\sigma \circ L_1) \circ L_2$ is always defined provided both $L_1$ and $L_2$ are not contradictory.

**Definition 2.2.** If $\sigma$ and $\sigma'$ are two global stores and there is a non-contradictory set $L$ of literals such that $\sigma' = \sigma \circ L$, then $\sigma'$ is called an