Information and Computation ••• (••••) •••–•••

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Distinguishing between communicating transactions ☆

Vasileios Koutavas *, Maciej Gazda, Matthew Hennessy

School of Computer Science and Statistics, Trinity College Dublin, Dublin, Ireland

ARTICLE INFO

Article history:
Received 7 February 2017
Received in revised form 3 September 2017
Available online xxxx

Keywords:
Non-isolated transactions
Communicating transactions
Hennessy–Milner logic
Bisimulation

ABSTRACT

Communicating transactions is a form of distributed, non-isolated transactions which provides a simple construct for building concurrent systems. In this paper we develop a logical framework to express properties of the observable behaviour of such systems. This comprises three nominal modal logics which share standard communication modalities but have distinct past and future modalities involving transactional commits. All three logics have the same distinguishing power over systems because their associated weak bisimulations coincide with contextual equivalence. Furthermore, they are equally expressive because there are semantics-preserving translations between their formulae. Using the logics we can clearly exhibit subtle example inequivalences. This work presents the first property logics for non-isolated transactions.

1. Introduction

Transactional constructs without the isolation principle have been proposed as useful building blocks of distributed systems (e.g., [10,11,17,23,3,6]). *Communicating transactions* is such a construct, equipped with a rich theory providing techniques for proving behavioural equivalence of transactional systems [8,9,16]. To develop useful verification tools, however, it is also essential to have techniques for exhibiting the *in*-equivalence of systems, rather than relying on the absence of equivalence proofs.

Numerous existing verification tools accept two formal descriptions of computing systems and determine whether or not they are behaviourally equivalent (e.g., [13,2,5]), and, crucially, provide coherent explanations as to why two descriptions are behaviourally distinguishable. Perhaps the most widely cited example is the relationship between the property language HML and the behavioural equivalence called *bisimulation equivalence* for processes written in the language *CCS*, [18]. Two processes are not equivalent, $P \not\approx Q$, if and only if there is an HML property ϕ which P enjoys and Q does not, [18,14]. Thus ϕ can be considered an explanation as to why P and Q have different behaviour. Indeed an algorithm has been proposed by Cleveland [4] and implemented in the *concurrency workbench* [5] which, when presented with descriptions of two finite state processes, either calculates a bisimulation, a formal justification for their behavioural equivalence, or returns a distinguishing HML formula.

For example consider the CCS process $P_0 = a.(b.\mathbb{0} + c.\mathbb{0})$, which performs an a-action, followed by offering a choice between a b- and a c-action, after which it terminates. According to the definition of bisimulation equivalence, $P_0 \not\approx Q_0$, where Q_0 denotes the slightly different process $a.(b.\mathbb{0}+c.\mathbb{0})+a.b.\mathbb{0}$. Intuitively p_1 satisfies the property: *whenever it performs*

☆ This work was supported by the Science Foundation Ireland grant 13/RC/2094 and co-funded under the European Regional Development Fund through the Southern & Eastern Regional Operational Programme to Lero – the Irish Software Research Centre.

* Corresponding author.
E-mail addresses: Vasileios.Koutavas@scss.tcd.ie (V. Koutavas), gazdam@scss.tcd.ie (M. Gazda), Matthew.Hennessy@scss.tcd.ie (M. Hennessy).

https://doi.org/10.1016/j.ic.2017.12.001
0890-5401/

Please cite this article in press as: V. Koutavas et al., Distinguishing between communicating transactions, Inf. Comput. (2017), https://doi.org/10.1016/j.ic.2017.12.001

an a-action it must be subsequently able to perform a c-action; whereas p_2 does not. In HML this property is captured by using modality operators, $[a]$ for necessity and $\langle a\rangle$ for possibility. Thus the property distinguishing P_0 from Q_0 is written formally as $[a]\langle c\rangle\,\mathtt{true}$.

The purpose of this paper is to develop similar property logics which characterise contextual equivalence for communicating transactions. As a formalism we use the abstract language *TCCS*^{m}, for which a natural contextual equivalence has been defined and characterised using a form of weak bisimulation over *configurations*, run-time entities recording the current state of the transactional system, together with information on historical interactions with its environment [16].

The transactional language *TCCS*^{m} is obtained by adding to *CCS* constructs for describing transactions. For example $P_1 = [\![a.b.\,\mathtt{co} \rhd_l d.\mathbf{0}]\!]$ describes a transaction named l which can either perform the sequence of actions a, b in its entirety, or else fails and performs the action d. The transaction $Q_1 = [\![a.(b.\,\mathtt{co} + c.\mathbf{0}) \rhd_l d.\mathbf{0}]\!]$ is a slight variation in which there is an apparent possibility of performing a c-action after a. However if this c-action is performed then the transaction can never commit (i.e., perform a co-action) and therefore the presence of this potential c-action is superfluous. According to *TCCS*^{m} reduction barbed equivalence theory [16] these two transactions are behaviourally equivalent. Consequently an extension of HML we propose should not be able to distinguish them, despite the fact that Q_1 can apparently perform a c-action or at least attempt to do so.

The notion of weak bisimulation developed in [16] for transactions contains constraints on the actions which transactions may perform—the standard *transfer property*. This in effect compares the future behaviour of transactions. But the definition of bisimulation also contains constraints on past behaviour, as encoded in configurations. For example, $P_2 = [\![a.\,\mathtt{co} \rhd_{k_1} \mathbf{0}]\!] \mid [\![b.\,\mathtt{co} \rhd_{k_2} \mathbf{0}]\!]$ can perform actions a and b and reach a state where these actions can be committed independently. On the other hand $Q_2 = \nu p.\,[\![a.p.\,\mathtt{co} + a.\,\mathtt{co} \rhd_{k_1} \mathbf{0}]\!] \mid [\![b.\overline{p}.\,\mathtt{co} + b.\,\mathtt{co} \rhd_{k_2} \mathbf{0}]\!]$ can perform the same actions and then, through the internal communication on p, can only commit the past actions a and b simultaneously. Thus two configurations reachable starting from P_2 and Q_2 are $\mathcal{C}_1 = \langle l_1(a), l_2(b) \bullet [\![\mathtt{co} \rhd_{l_1} \mathbf{0}]\!] \mid [\![\mathtt{co} \rhd_{l_2} \mathbf{0}]\!]\rangle$ and $\mathcal{C}_2 = \langle l(a), l(b) \bullet [\![\mathtt{co} \rhd_l \mathbf{0}]\!] \mid [\![\mathtt{co} \rhd_l \mathbf{0}]\!]\rangle$, respectively. In the latter configuration, the two transactions have been *merged* and thus obtained the same name l. After a single commit, C_1 becomes $\langle a, l_2(b) \bullet \mathbf{0} \mid [\![\mathtt{co} \rhd_{l_2} \mathbf{0}]\!]\rangle$ where only past action a is committed. Because there is no matching future configuration of $\mathcal{C}_2$, weak bisimulation from [16] distinguishes the two processes. Thus, to distinguish P_2 from Q_2, one would expect a property logic for transactions containing, in addition to standard future-oriented modal operators discussed above, operators for examining past behaviour.

In this paper we provide two such property logics, with different past operators. We also provide a property logic with *no* past operators; instead a richer collection of future-oriented operators are used. In the example of P_2 and Q_2 above, the first logic, $\mathcal{L}_{\mathsf{Hasco}}$, using only an additional "has committed" predicate on past actions ($\mathtt{Hasco}(k)$) can express the inequivalence as the following rather involved formula satisfied by Q_2:

$$\langle x(a)\rangle\,\langle y(b)\rangle\,\big(\neg\mathtt{Hasco}(x) \;\wedge\; \langle\tau\rangle\,\mathtt{Hasco}(x) \;\wedge\; ([\tau](\mathtt{Hasco}(x) \leftrightarrow \mathtt{Hasco}(y)))\big)$$

This states that the process can perform an a-action followed by a b-action and reach a state where: (1) the a-action has not been committed yet; (2) the a-action can be committed after some internal (τ) transitions; and (3) in any future configurations reachable by τ-transitions, the past a- and b-actions are either both committed or both aborted. Note that $\leftrightarrow$ is double implication, and x and y are bound variables representing the transactions performing a and b, respectively.

The second logic, $\mathcal{L}_{\mathsf{Eq}}$, distinguishes P_2 from Q_2 by the significantly simpler formula $\langle x(a)\rangle\,\langle y(b)\rangle(x =_{\mathtt{co}} y)$ which expresses the possibility of performing actions a and b, reaching a state where both have been committed by a single transaction, possibly as a result of transactional merging. The last logic, $\mathcal{L}_{\mathsf{Canco}}$, distinguishes the same processes with the formula $\langle x(a)\rangle\,\langle y(b)\rangle\,\langle \mathtt{co}(\{x, y\})\rangle\,\mathtt{true}$, expressing the possibility of performing a, then b, and then committing both actions simultaneously.

The main results of the paper include:

- Three property logics for *TCCS*^{m}, and their natural associated bisimulation relations. The first logic encapsulates the intuitions on observable past actions from [16]; the second encodes a more powerful predicate on past actions which we use to write more succinct formulas; the third logic uses only future action modalities giving rise to standard bisimulation equivalence. All logics include nominal [21,12] versions of the standard HML modal operators, and are based on a novel labelled transition system for *TCCS*^{m}.
- Proofs that all logics have the same distinguishing power over *TCCS*^{m} terms. In effect each of their associated bisimulations precisely coincides with the natural contextual equivalence.
- Proofs that each of the logics are *equally expressive*. We provide translations between the formulas of the three logics and show that any property definable in one logic is also expressible in each of the other two.

The remainder of the paper is organised as follows. First in Section 2 we recall the theory of *TCCS*^{m} from [16], in particular recalling the definition of bisimulation over *configurations* which characterises the natural contextual equivalence for transactions. Then in Section 3 we first explain the expressive deficiencies in this notion of configuration; that is the limited access it gives to past behaviour. We then propose a more expressive notion of *extended configuration*, together with a new notion of bisimulation, Hasco-bisimulation, over these extended configurations. This is similar in style to that in [16]; a transfer condition between possible actions puts requirements on the future behaviour of processes, while predicates

Please cite this article in press as: V. Koutavas et al., Distinguishing between communicating transactions, Inf. Comput. (2017), https://doi.org/10.1016/j.ic.2017.12.001