The decidability of hereditary history preserving bisimilarity on trace-labelled systems is unresolved

S. B. Fröschle
LFCS, Division of Informatics,
University of Edinburgh, Scotland
August 15, 2004

Abstract

In [1] it is presented that hereditary history preserving bisimilarity is decidable for finite trace-labelled systems. In this note we expose a gap in the argumentation, and demonstrate that the proof cannot easily be repaired.

In [1] it is presented that hereditary history preserving bisimilarity (hhp-b) is decidable for finite trace-labelled systems. In this note we expose a gap in the argumentation, and demonstrate that the proof cannot easily be repaired. In the following, we directly refer to [1] and assume the definitions therein. We also assume the standard game characterization of hhp-b [2].

A gap. [1] suggests that on trace-labelled systems hhp-b is captured by allowing Spoiler to backtrack within concurrent steps only. A corresponding concept step bisimilarity on step transition systems is introduced and shown to be equivalent with hhp-b via the intermediate notion of ‘two nets admit a synchronized step transition system’. However, the proof of ‘If two trace-labelled nets admit a synchronized step transition system then there exists a step bisimulation between the run foldings of the two nets with appropriate trace-relabellings.’ is incomplete. Consider the following statement, page 11 of [1], Proof of Lemma 4 ($\Rightarrow$):

“The fact that $R$ is a step bisimulation between $TS_1$ and $TS_2$ follows immediately from the definition of a synchronized step transition system for $N_1$ and $N_2$.”

We explain that this implication is, in fact, not immediate. Let us try to verify clause (iv) of the definition of step bisimulation. Let $((M_1, A_1), (M_2, A_2)) \in$
R and \((M'_1, A'_1), (M'_2, A'_2)\) ∈ R such that \((M'_1, A'_1) = \delta_1((M_1, A_1), u_1)\) and \((M'_2, A'_2) = \delta_2((M_2, A_2), u_2)\) for some \(u_1, u_2\) such that \(\lambda_1(u_1) = \lambda_2(u_2) = u\). We need to show: (*) for each \(v \subseteq u\), \((\delta_1((M_1, A_1), \lambda_1^{-1}(v)), \delta_2((M_2, A_2), \lambda_2^{-1}(v))) \in R\). We would like to have: (** \((r_1, r_2), (r'_1, r'_2) \in Q\) such that for \(i \in \{1, 2\}, M_i = M_{r_i}, A_i = top_{r_i}, M'_i = M'_{r_i}, A'_i = top'_{r_i}\), and \(\delta((r_1, r_2), (u_1, u_2)) = (r'_1, r'_2)\). Then, (*) would follow from clause (ii) of the definition of step transition system (applied to TS). By definition of R we have: \((r_1, r_2) \in Q\) such that for \(i \in \{1, 2\}, M_i = M_{r_i}\) and \(A_i = top_{r_i}\); and similarly \((r'_1, r'_2) \in Q\) such that for \(i \in \{1, 2\}, M'_i = M'_{r_i}\) and \(A'_i = top'_{r_i}\).

We argue it is not obvious that \(\delta((r_1, r_2), (u_1, u_2)) = (r'_1, r'_2)\). It is clear that there may be many runs \(r''_i \neq r'_i\) such that \(M_{r''_i} = M_{r_i}\) and \(top_{r''_i} = top_{r_i}\), \(i \in \{1, 2\}\). (For a trivial example, assume a net consisting of a loop of \(a\)-transitions and consider the two runs \(aaa\) and \(a\).) Then, it is also clear: although \(\delta_1(r_1, u_1) = r''_1\) and \(\delta_2(r_2, u_2) = r''_2\) for some \(r''_1, r''_2\) such that for \(i \in \{1, 2\}\), \(M_{r''_i} = M_{r_i}\), \(top_{r''_i} = top_{r_i}\), we do not necessarily have \(r''_1 = r'_1\) and \(r''_2 = r'_2\).

Hence, we cannot conclude \((r'_1, r'_2) \in Q\). There will be a match for \(u_1\) and \(u_2\) at \((r_1, r_2)\) in ST, but maybe \(u_1\) and \(u_2\) are not matched against each other at this point but by other transitions enabled at \(r_2\), and \(r_1\) respectively.

If \(ST\) is obtained from a hlp-bisimulation, we can deduce there must be \((r''_1, r''_2) \in Q\) such that \(\delta((r''_1, r''_2), (u_1, u_2)) = (r'_1, r'_2)\). We do have \(M_{r''_i} = M_{r_i}\), but not necessarily \(top_{r''_i} = A_{r_i}, i \in \{1, 2\}\). (For example, consider \(r''_1 = \varepsilon, r'_1 = a,\) and \(r_1 = aaa\) with respect to the above trivial example.) In other words, \((u_1, u_2)\) is indeed a match to reach state \((r'_1, r'_2)\), but we do not know whether \((r'_1, r'_2)\) is reached by \((u_1, u_2)\) from a state that translates into \(((M_1, A_1), (M_2, A_2))\).

To obtain (**\(i\)) it would be sufficient to prove that whenever \((r_1, r_2), (r''_1, r''_2) \in Q\) such that for \(i \in \{1, 2\}\), \(M_{r_i} = M_{r''_i}\) and \(top_{r_i} = top_{r''_i}\), then any \(u_1\) enabled at \(r_1\) (or \(r''_1\) equivalently) is matched at \((r_1, r_2)\) in exactly the same way as it is matched at \((r''_1, r''_2)\), and symmetrically for \(u_2\). But this is a strong requirement. The undecidability of hlp-b shows that it is not given in the general case. One needs a further argument why this should apply for trace-labelled systems.

**Discussion and Counter-example.** The above problem seems to originate from the fact that step bisimilarity does not adequately capture the idea of allowing Spoiler to backtrack within concurrent steps only. Step bisimilarity is very strong in that it involves backtracking to positions that are not necessarily reachable by ‘shuffling’ the matching that is laid down during a play. A revised version of step bisimilarity, *history preserving bisimilarity plus backtracking within matched steps (bstep-hp-b)*, is captured in terms of games as follows.

We define the *bstep-hp-b game* between Spoiler and Duplicator directly for two nets \(N_1, N_2\). Configurations are pairs \((r_1, r_2) \in Runs(N_1) \times Runs(N_2)\); we assume the runs to be structured into concurrent steps as laid down by a play. The initial configuration is \((\varepsilon, \varepsilon)\). A play is played as follows: Spoiler chooses one of \(N_1\) or \(N_2\), say \(N_1\), and performs a concurrent step \(u_1\) that is enabled at \(r_1\). Duplicator has to respond by executing a concurrent step \(u_2\) that is enabled
at \( r_2 \) such that \( r_1u_1 \simeq r_2u_2 \). At this point, and only at this point, Spoiler is allowed to backtrack any set of transitions \( v_1 \subseteq u_1 \) in \( r_1u_1 \); Duplicator has to backtrack the corresponding transitions in \( r_2u_2 \), and play resumes at the new position. Play continues like this forever, in which case Duplicator wins, or until either Spoiler or Duplicator is unable to move, in which case the other participant wins.

It is routine to check that hhp-b does imply btstep-hp-b. However, the counter-example of Figure 1 and 2 demonstrates the opposite direction is now lost: btstep-hp-b does not imply hhp-b, not even for trace-labelled systems. The behaviour of \( S \) and \( S' \) is consistent with the following trace alphabet: \( eIf, eDg, fDg, eDi, fDj \), and the obvious relations for the remaining labels. To win the hhp-b game Spooner performs \( ce_1bf_2 \): after \( ce_1 \) there is the potential of \( i \), which forces Duplicator to match \( e_1 \) by \( e'_1 \); after \( ce_1bf_2 \) transition \( g_1 \) is enabled, and Duplicator has to match \( f_2 \) by \( f'_2 \). But then Spooner can backtrack \( e_1 \) and \( c \), perform \( d \), and expose the \( j \). To see that \( S \) and \( S' \) are btstep-hp-bisimilar, consider that to detect the difference between \( S \) and \( S' \), Spooner needs to backtrack over two causally dependent transitions such as \( bf_2 \). But this is not possible in the btstep-hp-b game. In more detail, Duplicator can win the btstep-hp-b game as follows.

Duplicator makes her strategy dependent on the first step in the configura-
tion of the play. We group first steps into three groups: (1) \{a, d\}; (2) a, c, \{a, c\}; (3) d, b, \{d, b\}; (4) \{b, c\}. The matching of these steps is forced by the labelling. Backtracking within them will bring us back to configurations already covered by (1) to (4). For subsequent steps Duplicator proceeds according to the group of the first step in the current configuration: (1) There is nothing more to match. (2) Duplicator orientates herself by the \(i\)- and \(g\)-synchronizations, and will match the \(e\)- and \(f\)-transitions as follows: \(e_1-e_1', e_2-e_2', f_1-f_2, f_2-f_1'\). Duplicator is in no danger from \(j\) since either the \(j\)'s, or the \(d\), or both are disabled now and at any subsequent point in the game. (3) Symmetrically, Duplicator orientates herself by the \(j\)- and \(g\)-synchronizations, and matches: \(e_1-e_2', e_2-e_1', f_1-f_1', f_2-f_2'\). (4) Duplicator commits herself to either strategy; both of them will win the game. Backtracking within these steps will bring us back to a configuration where the same strategy applies (since the first step will remain unaltered). This guarantees that Duplicator’s strategy is always winning in the btstep-hp-b game.

**Conclusion.** Hhp-b may well be decidable for finite trace-labelled systems. But unfortunately, it appears we will require a technique different to the one suggested in [1].

**Acknowledgements.** I thank Madhavan Mukund for checking the argumentation of this note.
References
