ECAI 2010
H. Coelho et al. (Eds.)
IOS Press, 2010

doi:10.3233/978-1-60750-606-5-1037

Trust in complex actions

Julien Bourdon[1] and **Guillaume Feuillade** and **Andreas Herzig** and **Emiliano Lorini**[2]

1 Introduction

Recently Herzig, Lorini et col. undertook a logical analysis of the definition of trust proposed by Castelfranchi and Falcone (C&F henceforth) [3, 4]. They defined the predicate $\texttt{Trust}_0(i, j{:}a, \varphi)$ (agent i trusts agent j to do action a in order to achieve i's goal φ) as follows:[3]

$$\texttt{Goal}(i,\varphi) \wedge \texttt{Bel}_i(\texttt{CExt}(i{:}a) \wedge \texttt{CInt}(i{:}a) \wedge \texttt{Res}(i{:}a,\varphi))$$

$\texttt{Goal}(i,\varphi)$ means that i has the goal that φ; $\texttt{CExt}(i{:}a)$ are the external conditions (conditions on the world): j is capable to perform a; $\texttt{CInt}(i{:}a)$ are the internal conditions (conditions on the agent's mental state): j is willing to perform a; and $\texttt{Res}(i{:}a,\varphi)$ links the conditions on the agent's mental state with those on the world: j has the power to achieve φ by doing a.[4] They then defined the predicates $\texttt{CExt}(j{:}a)$, $\texttt{CInt}(j{:}a)$ and $\texttt{Res}(j{:}a,\varphi)$ in terms of the concepts of belief, choice, action and time. Both C&F and Herzig, Lorini et col. only considered trust in the atomic action of another agent and did not consider trust in complex actions where the elements in the complex action are atomic actions of different agents.

In the context of services architecture, some provider agents publish atomic services; however, when a user needs to implement a more complex business process, it must chain service calls according to a specific workflow structure. Automating the service calls is called service composition: given the business process to implement, the control flow has to be computed in order to guarantee that the goal of the service caller is satisfied. Since services are provided by agents, users may trust some agents for certain actions but not for other actions which they deem critical, usually depending on the nature of the information they have to send to this agent for the service action to perform the action. In the literature (e.g. [5]), service selection for composition assumes the existence of a central authority guaranteeing the non-functional properties of the services. In practice such an authority might not exist (cf. e.g. P2P networks [6]), or may not itself be trustworthy. A model for trust in the services world, allows to express composition objectives as dynamic logic formulas with a trust component. The service composition problem would then reduce to the satisfaction of such a formula. This method would ensure that the composition is correct and compatible with the beliefs of the user, thus ensuring a trustworthy sequence of service call for achieving the goal of the user.

The details are presented in the extended version of the paper [1].

2 A logical language with complex actions

We recall the logical framework of [3, 4], that we extend towards complex actions. Given a set of propositional variables *Atm*, a set of agents *Agt* and a set of atomic actions *Act*, complex formulas φ and complex actions α are defined by the following BNF:

$$\begin{array}{lcl} \varphi & ::= & p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \texttt{Bel}_i\varphi \mid \texttt{Ch}_i\varphi \mid \texttt{Feasible}_\alpha\varphi \mid \\ & & \texttt{Happens}_\alpha\varphi \mid \texttt{F}\varphi \\ \alpha & ::= & i{:}a \mid \alpha;\alpha \mid \alpha{+}\alpha \mid \varphi? \mid \alpha^* \end{array}$$

where p ranges over *Atm*, i ranges over *Agt*, and a ranges over *Act*. $\texttt{Bel}_i\varphi$ reads "i believes that φ"; $\texttt{Ch}_i\varphi$ reads "i chooses that φ"; $\texttt{Feasible}_\alpha\varphi$ reads "there is a possible execution of α after which φ is true"; $\texttt{Happens}_\alpha\varphi$ reads "α happens, and φ is true afterwards"; and $\texttt{F}\varphi$ reads "φ will eventually be true".

The atomic action $i{:}a$ reads "i performs a"; the complex action $\alpha_1;\alpha_2$ reads "do α_1 and then α_2"; $\alpha_1{+}\alpha_2$ reads "choose nondeterministically between α_1 and α_2", where the choice is understood to be up to the environment, and not up to the agents performing α_1 and α_2; $\varphi?$ reads "if φ is true then continue, else fail"; and finally, α^* reads "do α an arbitrary number of times".

We define $\texttt{After}_\alpha\varphi$ to be an abbreviation of $\neg\texttt{Feasible}_\alpha\neg\varphi$. Moreover we have the following standard program constructions:

$$\begin{array}{rcl} \texttt{skip} & \stackrel{\text{def}}{=} & \top? \\ \texttt{fail} & \stackrel{\text{def}}{=} & \bot? \\ \texttt{if}\ \varphi\ \texttt{then}\ \alpha_1\ \texttt{else}\ \alpha_2 & \stackrel{\text{def}}{=} & (\varphi?;\alpha_1){+}(\neg\varphi?;\alpha_2) \\ \texttt{while}\ \varphi\ \texttt{do}\ \alpha & \stackrel{\text{def}}{=} & (\varphi?;\alpha)^*;\neg\varphi? \end{array}$$

3 Trust about complex actions

We now generalize the definition of (occurrent) trust about atomic actions of [3, 4] to trust about complex actions and study its constituents. Among all possible complex actions we here only consider deterministic actions [2]: actions built with "skip", "fail", ";", "if φ then α_1 else α_2", and "while φ do α". Their BNF is:

$$\begin{array}{lcl} \alpha & ::= & i{:}a \mid \texttt{skip} \mid \texttt{fail} \mid \\ & & \alpha;\alpha \mid \texttt{if}\ \varphi\ \texttt{then}\ \alpha\ \texttt{else}\ \alpha \mid \varphi? \mid \texttt{while}\ \varphi\ \texttt{do}\ \alpha \end{array}$$

We therefore do not consider nondeterministic composition and iteration in our analysis of trust in complex actions. Note that tests $\varphi?$ can be defined as $\texttt{if}\ \varphi\ \texttt{then skip else fail}$.

Let us first recall the definition of the original trust predicate in [3, 4]. There, the goal condition $\texttt{Goal}(i,\varphi)$ was defined as $\texttt{Ch}_i\texttt{F}\varphi$, i.e. as i's choice of futures where φ holds. The external condition $\texttt{CExt}(j{:}a)$

[1] Kyoto University, Department of Social Informatics, Japan
[2] Université de Toulouse, CNRS, IRIT, France

[3] They distinguish occurrent trust —i's trust that j is going to perform a here and now— and dispositional trust: i's trust that j is going to perform a whenever some suitable conditions obtain. We here consider the former.

[4] They used a 4-ary predicate $\texttt{Trust}(i,j,a,\varphi)$ instead of our ternary $\texttt{Trust}_0(i,j{:}a,\varphi)$. Moreover, their trust definition was in terms of the predicates *Capable*$(j{:}a)$, *Willing*$(j{:}a)$ and *Power*$(j{:}a,\varphi)$ instead of our $\texttt{CExt}(j{:}a)$, $\texttt{CInt}(j{:}a)$ and $\texttt{Res}(j{:}a,\varphi)$. We here introduce these terms and notations because they better generalize to complex actions.

was defined as $\texttt{Feasible}_{j:a}\top$ (j:a is executable), and the internal condition $\texttt{CInt}(j{:}a)$ as $\texttt{Ch}_j\texttt{Happens}_{j:a}\top$ (j chooses that j:a is going to occur). Finally, the power condition $\texttt{Res}(j{:}a,\varphi)$ was defined as $\texttt{After}_{j:a}\varphi$ (φ will hold immediately after every possible performance of j:a).

3.1 Definition of trust

Our move from trust in atomic actions to trust in complex actions requires an adjustment of the definition of trust in a complex action:

$$\texttt{Trust}(i,\alpha,\varphi) \stackrel{\text{def}}{=} \texttt{Goal}(i,\varphi)\wedge\texttt{Bel}_i(\texttt{CExt}(\alpha)\wedge\texttt{CInt}(\alpha)\wedge\texttt{Res}(\alpha,\varphi))$$

where i is an agent, α is a deterministic action, and φ is a formula. As before, CExt and CInt stand for the external and the internal conditions in trust assessment.

Observe that trust in atomic actions involved a single trustee j. Here we have to account for trust in complex actions that may be performed by several agents; we therefore consider trust in a group of agents. Note also that before, the trustee j —which here would be a set of agents J— appeared explicitly in the definition of the predicate Trust. However, one may consider that J is implicitly already there: it is the set of agents occurring in α. Therefore the agent argument need not to appear as a separate argument in the definition. It remains to explain the predicates on the right hand side of the definition of trust.

3.2 Defining the ingredients of trust

The definition of the Goal predicate transfers straightforwardly because no action occurs in it:

$$\texttt{Goal}(i,\varphi) \stackrel{\text{def}}{=} \texttt{Ch}_i\texttt{F}\varphi$$

So it remains to define CExt, CInt and Res.

The original power condition $\texttt{Bel}_i\texttt{After}_{j:a}\varphi$ stipulated that i believes φ immediately results from j's performance of atomic action a. However, consider i's trust in j_1 and j_2 to perform the sequence of actions $j_1{:}a_1; j_2{:}a_2$ in order to achieve i's goal φ. With respect to which goal should i trust j_1? The truster i typically does not bother about the direct effect of j_1's action a_1 and is only interested in the overall effect φ of the complex action $j_1{:}a_1; j_2{:}a_2$. In other words, we have to account for the case where φ is not achieved immediately, but only at some time point in the future. We therefore redefine

$$\texttt{Res}(\alpha,\varphi) \stackrel{\text{def}}{=} \texttt{After}_\alpha\texttt{F}\varphi$$

Up to now, all our definitions were directly in terms of well-defined formulas of our logic. Things are not as simple for the external condition CExt and for the internal condition CInt.

As to the external condition, $\texttt{CExt}(\alpha)$ means that the complex action α is executable *whatever the other agents and nature choose to do*. This means that the preconditions of α must obtain at every step of every execution of α. It follows that while $\texttt{CExt}(\alpha)$ implies $\texttt{Feasible}_\alpha\top$, it should not be equivalent to it. For example, the complex action $(i{:}a{+}i{:}a'); i{:}b$ cannot be said to be executable (in the above sense) when just $\texttt{Feasible}_{i:a+i:a';i:b}\top$ holds. Indeed, a situation where $\texttt{Feasible}_{i:a'}\texttt{After}_{i:b}\top$ is compatible with the latter formula, and if nature chooses $i{:}a'$ when executing the nondeterministic $i{:}a{+}i{:}a'$ then it cannot be said that $\texttt{Feasible}_{i:a+i:a';i:b}\top$ is executable.

Given these considerations we recursively define $\texttt{CExt}(\alpha)$ as:

$$\begin{aligned}
\texttt{CExt}(\texttt{skip}) &\stackrel{\text{def}}{=} \top & \texttt{CExt}(\texttt{fail}) &\stackrel{\text{def}}{=} \bot\\
\texttt{CExt}(i{:}a) &\stackrel{\text{def}}{=} \texttt{Feasible}_{i:a}\top\\
\texttt{CExt}(\alpha;\beta) &\stackrel{\text{def}}{=} \texttt{CExt}(\alpha)\wedge\texttt{After}_\alpha\texttt{CExt}(\beta)\\
\texttt{CExt}(\texttt{if }\varphi\texttt{ then }\alpha_1\texttt{ else }\alpha_2) &\stackrel{\text{def}}{=} (\varphi\wedge\texttt{CExt}(\alpha))\vee(\neg\varphi\wedge\texttt{CExt}(\beta))\\
\texttt{CExt}(\texttt{while }\psi\texttt{ do }\alpha) &\stackrel{\text{def}}{=} \texttt{F}\neg\psi\wedge\texttt{After}_{(\psi?;\alpha)^*;\psi?}\texttt{CExt}(\alpha)
\end{aligned}$$

The clause for ";" makes that $\texttt{CExt}(\alpha)$ is stronger than $\texttt{Feasible}_\alpha\top$. As to the (internal) willingness condition, we define $\texttt{CInt}(\alpha)$ as:

$$\begin{aligned}
\texttt{CInt}(\texttt{fail}) &\stackrel{\text{def}}{=} \top & \texttt{CInt}(\texttt{skip}) &\stackrel{\text{def}}{=} \top\\
\texttt{CInt}(i{:}a) &\stackrel{\text{def}}{=} \texttt{Ch}_i\texttt{Happens}_{i:a}\top\\
\texttt{CInt}(\alpha;\beta) &\stackrel{\text{def}}{=} \texttt{CInt}(\alpha)\wedge\texttt{After}_\alpha\texttt{CInt}(\beta)\\
\texttt{CInt}(\texttt{if }\varphi\texttt{ then }\alpha_1\texttt{ else }\alpha_2) &\stackrel{\text{def}}{=} (\varphi\wedge\texttt{CInt}(\alpha))\vee(\neg\varphi\wedge\texttt{CInt}(\beta))\\
\texttt{CInt}(\texttt{while }\psi\texttt{ do }\alpha) &\stackrel{\text{def}}{=} \texttt{After}_{(\psi?;\alpha)^*;\psi?}\texttt{CInt}(\alpha)
\end{aligned}$$

4 Properties of trust

First of all, we observe that our and the original definition coincide for atomic actions, except that we have relaxed the result condition: for us it suffices that the result φ obtains at some point in the future, and not immediately after the action. We therefore have $\texttt{Trust}(i,j{:}a,\varphi)\leftrightarrow\texttt{Trust}_0(i,j{:}a,\texttt{F}\varphi)$.

For complex actions we are going to have reductions in terms of equivalences for the cases of skip, fail, if-then-else conditionals and while loops. For trust in sequential compositions we only give a sufficient condition.

Theorem 1 *The following formulas are valid.*

- $\texttt{Trust}(i,\texttt{fail},\varphi)\leftrightarrow\bot$
- $\texttt{Trust}(i,\texttt{skip},\varphi)\leftrightarrow(\texttt{Goal}(i,\varphi)\wedge\texttt{Bel}_i\texttt{F}\varphi)$
- $(\texttt{Trust}(i,\alpha,\varphi)\wedge\texttt{Bel}_i\texttt{After}_\alpha\texttt{Trust}(i,\beta,\varphi))\rightarrow\texttt{Trust}(i,(\alpha;\beta),\varphi)$
- $\texttt{Trust}(i,(\alpha;\beta),\varphi)\rightarrow\texttt{Trust}(i,\alpha,\varphi)$
- $\texttt{Trust}(i,(\alpha;\beta),\varphi)\rightarrow\texttt{After}_\alpha(\neg\texttt{Goal}(i,\varphi)\vee\texttt{Trust}(i,\beta,\varphi))$
- $\texttt{Trust}(i,\texttt{if }\psi\texttt{ then }\alpha\texttt{ else }\beta,\varphi)\leftrightarrow$
  $\texttt{Bel}_i((\psi\rightarrow\texttt{Trust}(i,\alpha,\varphi))\wedge(\neg\psi\rightarrow\texttt{Trust}(i,\beta,\varphi)))$
- $\texttt{Trust}(i,(\texttt{while }\psi\texttt{ do }\alpha),\varphi)\leftrightarrow$
  $\texttt{Bel}_i\texttt{After}_{(\psi?;\alpha)^*}(\psi\rightarrow(\texttt{CExt}(\alpha)\wedge\texttt{CInt}(\alpha)))\wedge$
  $\texttt{Goal}(i,\varphi)\wedge\texttt{Bel}_i\texttt{After}_{\texttt{while }\psi\texttt{ do }\alpha}\texttt{F}\varphi\wedge\texttt{Bel}_i\texttt{F}\neg\psi$

REFERENCES

[1] J. Bourdon, G. Feuillade, A. Herzig, and E. Lorini. Trust in complex actions. In D. Gabbay and L. van der Torre, editors, *Proceedings of the International Workshop on Logics in Security 2010*, Copenaghen, 9-13 August, 2010. to appear.

[2] J.Y. Halpern and J. H. Reif. The propositional dynamic logic of deterministic, well-structured programs. *Theoretical Computer Science*, 27:127–165, 1983.

[3] Andreas Herzig, Emiliano Lorini, Jomi F. Hübner, and Laurent Vercouter. A logic of trust and reputation. *Logic Journal of the IGPL*, to appear.

[4] E. Lorini and R. Demolombe. Trust and norms in the context of computer security. In *Proc. of the Ninth International Conference on Deontic Logic in Computer Science (DEON'08)*, number 5076 in LNCS, pages 50–64. Springer-Verlag, 2008.

[5] E. Sirin, B. Parsia, and J. Hendler. Composition-driven filtering and selection of semantic web services. In *AAAI Spring Symposium on Semantic Web Services*, pages 129–138, 2004.

[6] Y. Wang and J. Vassileva. Trust and reputation model in peer-to-peer networks. In *Proc. of IEEE Conference on P2P Computing*, pages 150–157. IEEE Press, 2003.