IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 5, Ver. V (Sep. - Oct. 2016), PP 49-55
www.iosrjournals.org

Pi-calculus based Bayesian Trust Web Service Composition

Bensheng Yun, Yaguan Qian
(Department of Mathematics and Information Science, Zhejiang University of Science and Technology, China)

Abstract : *To enhance the reliability of trust Web service composition, Pi-calculus based formal verification of trust Web service composition is proposed. Bayesian trust Web service composition is firstly defined abstractly; then Pi-calculus is used to describe its composition structure and internal interaction, the mapping relation between trust entity and Pi-calculus is provided. The automatic reasoner MWB is adopted to manipulate and analyze the composition system, which is aimed at finding and correcting the faults before the implementation.*
Keywords: *formal verification, Pi-calculus, trust web service composition, MWB*

I. Introduction

Web service has become one of the most important computing resource, however, the network environment is dynamic, distributed, open, and uncertain, and so on. Since these features may result in many uncertain factors, it is badly in need of a secure and reliable management tool. A valid method is to evaluate trust value of the network entity and establish trust mechanism for Web service. Trust involves many factors, but it is generally acknowledged: Trust ≈ Security + Reliability [1]. Some evaluations of trust have been proposed in previous works, such as probability and statistic based method [2, 3, 4], fuzz based method [5].

Web service composition is the main interaction system. During the composition process, both sides of interactive Web services need to assess mutual trust value. Once every Web service satisfied with each other on trust value, the composition can be implemented further. With larger and larger scale of Web service composition, the trust authentication of Web service is becoming more and more complex and error-prone. Therefore, it will affect the credibility of software which based on Web service composition technology [6].

In recent years, some formal methods are used to analyze and verify Web service composition, such as Pi-calculus [7], Petri Net [8]. However, there lacks of describing and analyzing trust authentication. It is necessary to analyze and verify trust authentication of Web service composition, so that the errors can be found and corrected before the implementation. Pi-calculus owns the powerful behavior equivalence theory and itself is also in constant development, such as Pi^+-calculus [9]. Therefore, Pi-calculus is a useful tool for formal verification.

II. Bayesian Trust Web Service Composition

The service oriented network and the real human-centered social network have a high degree of similarity [10]. For Web service, the execution success probability reflects its reliability, and the ability provide resource as well. Which Web service has higher probability of the execution is more credible. Therefore, the execution success probability can be served as a measure of trust.

Bayes method is that subject uses former objective data information, according to its experience and knowledge, evaluates the probability of occurrence of the event, which are both objective and subjective [11]. So, this paper uses it to evaluate the execution success probability of Web service, i.e. measure its trust value.

- Direct trust

Let S_i and S_j are two Web service, S_i analyzes its invocation history with S_j, the trust from S_i's direct experiences is called *Direct Trust*, denoted by DT_{ij}.

Suppose S_i invokes S_j n times in the past time, where the execution successes u times and fails n-u times. Let p be the execution success probability of S_j, X be the sample, x be value of sample. Suppose that X obeys binomial distribution $B(n, p)$ since the p's prior distribution without information. According to Bayes supposition and conjugate prior distribution, $Beta(1, 1)$ is taken as p's prior distribution $\pi(p)$. since $\pi(p) = 1$, $(0 \le p \le 1)$, $f(x|p)= C_n^u p^u (1-p)^{n-u}$, the posterior density of p is

$$h(p \mid x) = \frac{f(x \mid p)\pi(p)}{\int_0^1 f(x \mid p)\pi(p)dp} = \frac{p^u(1-p)^{n-u}}{\int_0^1 p^u(1-p)^{n-u}dp} = \frac{p^u(1-p)^{n-u}}{\frac{(n-u)!u!}{(n+1)!}} = \frac{(n+1)!}{(n-u)!u!}p^u(1-p)^{n-u}$$

$$= \frac{\Gamma(n+2)}{\Gamma(n-u+1)\Gamma(u+1)}p^u(1-p)^{n-u} = Beta(u+1, n-u+1). \quad (1)$$

According to equation (1), p's posterior density obeys Beta distribution. Under the condition with "no information", maximum likelihood estimation is an excellent method [14]. So, at the $(n+1)th$ time invocation, the execution success probability can be defined as the maximum likelihood estimation of p. Thus

$$\ln h(p \mid x) = \ln \frac{\Gamma(n+2)}{\Gamma(n-u+1)\Gamma(u+1)} + u \ln p + (n-u)\ln(1-p),$$

$$\text{let} \quad \frac{\partial \ln h(p \mid x)}{\partial p} = \frac{u}{p} - \frac{n-u}{1-p} = 0, \text{ then} \quad p = \frac{u}{n}. \tag{2}$$

Definition 1. There exists two Web services: S_i and S_j. We will use binary event (success and failure) to describe the result of execution. Suppose S_i invokes S_j n times (where successes u times and fails n-u times). The direct trust value (DTV_{ij} for short) is defined as the execution success probability at the $(n+1)th$ time invocation. Then the execution success probability obeys Beta distribution, its maximum likelihood estimation is

$$DTV_{ij} = \frac{u}{n},\ 0 \le u \le n. \tag{3}$$

- Recommend Trust

If S_i only has no or limited direct invocation experiences with S_j, a natural way to get trust value for S_i is to ask its acquaintances about their opinions. S_i asks its one acquaintance, S_k, to get the trust value with S_j. The trust from acquaintance S_k is called *Recommend Trust*, denoted by RT_{ikj}. The direct trust value between S_k and S_j is denoted as DTV_{kj}, S_k recommends its direct experiences to S_i, and then these experiences become indirect experiences of S_i. But S_k may be not a very familiar friend for S_i, or S_k has recommend S_i inaccurate experiences in the past. Hence, S_i does not think S_k's recommendation is completely right. For example, S_i may say an 80 percent probability that S_k's recommendation is right. 80 percent shows the degree of S_k's recommendation for S_i. is called *Recommendation Degree*, which is denoted $R_{ik} \in [0, 1]$.

Definition 2. Suppose that there are three nodes of network: invokers S_i, S_k and service S_j. The direct trust value between S_j and S_k is $DT_{kj} \in [0, 1]$. The recommendation value is $R_{ik} \in [0, 1]$. Then the recommend trust value is defined as

$$RTV_{ikj} = R_{ik} \times DTV_{kj} \in [0, 1]. \tag{4}$$

Since trust is mutual, the identification of trust subject and trust object is relative, which depends on their environment. In the service oriented network, the evaluation of trust value mainly depends on their own experience and the third party's recommendation. In theory, a trust relation can be established between any entities in the network, such as A and B, which are denoted by $TR(A,B)$.

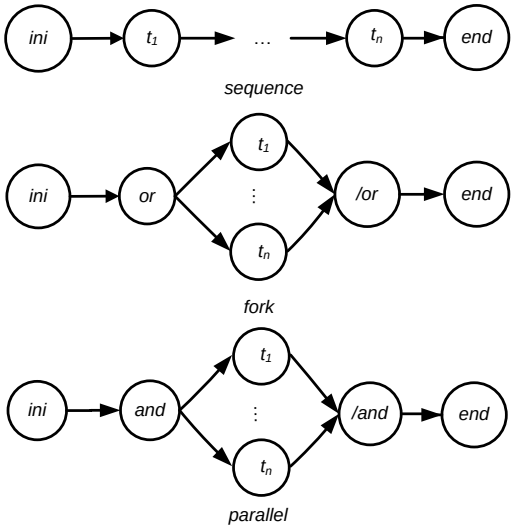

Figure 1. The basic control relations (*CR*)

Definition 3. (Bayesian Trust Web Service Composition) In the service oriented network, *Bayesian Trust Web Service Composition* can be defined as a three-tuple: $TWSC = <WS, CR, TR>$, where

- $WS = \{ WS_1, WS_2, \ldots, WS_n\}$ the set of trust entities, there are control relation and trust relation between entities;
- $CR = \{sequence, fork, parallel\}$ the set of basic control relations, as shown in Figure 1;
- $TR = \{direct\ trust, recommend\ trust\}$ the trust relation set between entities, the trust value is evaluated by Bayesian method.

An execution plan of trust Web service composition as shown in Figure 2, the set of trust entities in composition is $WS = \{WS_1, WS_2, WS_3, WS_4, WS_5, WS_6\}$, the set of basic control relations is $CR = \{sequence, and\}$, and the trust relation set is $TR = \{TR(WS_1,WS_2), TR(WS_1, WS_3), TR(WS_2, WS_4), TR(WS_3, WS_5), TR(WS_4, WS_6), TR(WS_5, WS_6)\}$.

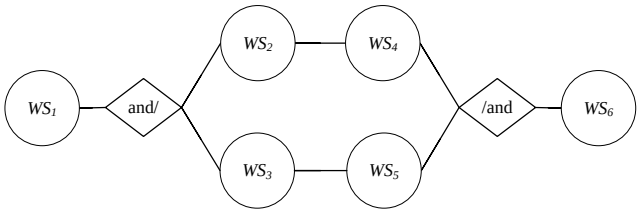

Figure 2. An execution plan of Web service composition

III. Pi-calculus

Pi-calculus is a process algebra for specifying and reasoning about concurrent systems. Although we refer to [12] for a detail description of Pi-calculus, a brief introduction to its syntax, transition relations and behavior equivalence theory is given as follows.

Definition 4. (Pi-calculus) The *processes* of the Pi-calculus are given respectively by

$$P ::= \mathbf{0} \mid \pi.P \mid P+Q \mid P|Q \mid \nu\tilde{z}\,P \mid !P \mid A(\tilde{x})$$

$$\pi ::= \bar{x}\tilde{y} \mid x(\tilde{y}) \mid \tau \mid [x = y]\,\pi.P .$$

- $\mathbf{0}$ is inaction; it is a process that can do nothing.
- The *prefix* $\pi.P$ has a single capability, expressed by π; the process P cannot proceed until that capability has been exercised.

  The *output prefix* $\bar{x}\tilde{y}.P$ can send the name tuple $\tilde{y}$ via the name x and continue as P. The *input prefix* $x(\tilde{y}).P$ can receive any name tuple via x and continue as P with the received name substituted for $\tilde{y}$. The *unobservable prefix* $\tau.P$ can evolve invisibly to P. τ can be thought of as expressing an internal action of a process.The *match prefix* $[x = y]\,\pi.P$ can evolve as $\pi.P$ if x and y are the same name, and can do nothing otherwise.
- The capabilities of the *sum* $P+Q$ are those of P together with those of Q. When a sum exercises one of its capabilities, the others are rendered void.
- In the *composition* $P|Q$, the components P and Q can proceed independently and can interact via shared names.
- In the restriction $\nu\tilde{z}\,P$, the scope of the name tuple $\tilde{z}$ is restricted to P.
- The *replication* $!P$ can be thought of as an infinite composition $P \mid P \mid \ldots$, replication is the operator that makes it possible to express infinite behaviors.
- The *process identifer* $A(\tilde{x})$, each process identifer can be defined as $A(\tilde{x}) = P$.

Definition 5. (Transition relations) The *transition relations* are defined by the rules in Table 1.

Table 1. The transition rules

$Act: \dfrac{-}{\pi.P \xrightarrow{\pi} P}$	$Match: \dfrac{\pi.P \xrightarrow{\pi} P}{[x = x]\pi.P \xrightarrow{\pi} P}$
$Sum_j: \dfrac{P_j \xrightarrow{\pi} P_j'}{\sum_{j\in I} P_j \xrightarrow{\pi} P_j'}$	$Res: \dfrac{P \xrightarrow{\pi} P'}{\nu\tilde{z}P \xrightarrow{\pi} \nu\tilde{z}P'}, (\tilde{z} \notin fn(P))$
$Par: \dfrac{P \xrightarrow{\pi} P'}{P \mid Q \xrightarrow{\pi} P' \mid Q}, (bn(\pi) \cap fn(Q) = \phi)$	
$Com: \dfrac{P \xrightarrow{\bar{x}\tilde{y}} P', Q \xrightarrow{x(\tilde{z})} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'\{\tilde{y}/\tilde{z}\}}$	$Id: \dfrac{P \xrightarrow{\pi} P'}{A \xrightarrow{\pi} P'}, (A \overset{def}{=} P)$

Definition 6. (Sequential composition) The *sequential composition* $P;Q$ means that 'when P finishes, Q starts'. Set $\bar{d}$ be the last action of process P,

$$P;Q = \nu d(\bar{d}.P \mid d(\ ).Q) .$$

Definition 7. (Weak equivalence) Let R be a binary relation over processes, then R is said to be a *weak simulation* if, whenever $(P, Q) \in R$,

If $P \overset{e}{\Rightarrow} P'$, then $\exists Q'$ s.t. $Q \overset{e}{\Rightarrow} Q'$ and $(P', Q') \in R$,

Where $e = \alpha_1 \alpha_2 \cdots \alpha_n$, $\overset{e}{\Rightarrow} = \overset{\tau}{\Rightarrow} \xrightarrow{\alpha_1} \overset{\tau}{\Rightarrow} \cdots \overset{\tau}{\Rightarrow} \xrightarrow{\alpha_n} \overset{\tau}{\Rightarrow}$, $\overset{\tau}{\Rightarrow} = \xrightarrow{\tau} \cdots \xrightarrow{\tau}$, the transitive reflexive closure of $\xrightarrow{\tau}$. R is said to be a *weak bi-simulation* if both R and its converse are weak simulations. P and Q are called *weakly bi-similar*, *weakly equivalent* or *observation equivalent*, if there exists a weak bi-simulation such that $(P, Q) \in R$, denoted by $P \approx Q$.

IV. Formal Model of Trust Web Service Composition

According to the similarities of TWSC and Pi-calculus, the rule of correspondence from TWSC to the Pi- calculus is established, as shown in table 2.

Table 2. Elements mapping between TWSC and Pi-calculus

TWSC		**Pi-calculus**	
Web service		Process	
Operation		Action	
Message		Message	
Communication		Interaction(τ)	
CR	Sequence	;	Operator
	Fork	+	
	Parallel	|	

Each trust entity is regarded as a process in Pi-calculus, the interaction between two trust entities is represented by τ action. The three control relations in composition: sequence, fork and parallel are mapping to ";", "+" and "|" respectively, which are included in Pi-calculus. However, how to identify trust entity (process) contained in composition, and how to identify the channel between the interactive trust entities. Therefore, three rules are proposed to identify process and channel.

- **Rule 1.** In the trust Web service composition, one trust entity (atomic Web service) corresponds to one process.
- **Rule 2.** In the trust Web service composition, two interactive trust entities share one channel at least logically.
- **Rule 3.** In the trust Web service composition, allowing multiple small trust entities to be combined to form a bigger trust entity, and a bigger trust entity can be divided into several small trust entities.

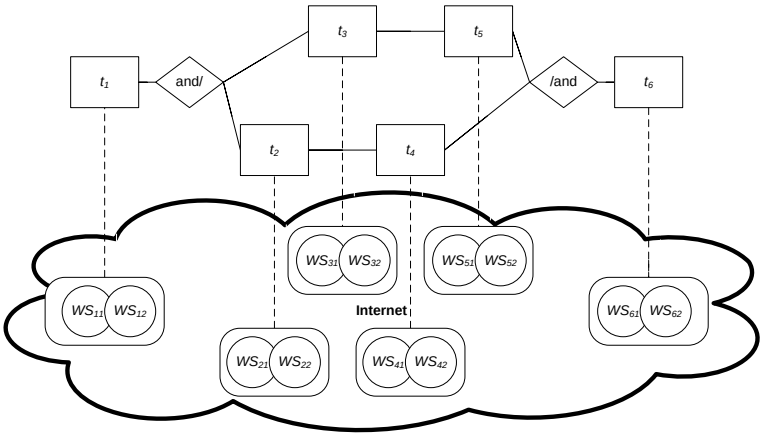

Figure 3. Trust Web service composition

As shown in Figure 3, each task node has two candidate Web services, according to Table 2, Rule 1 and 2, the interaction diagram between processes is presented in Figure 4.The trust certification between WS_{21} and WS_{41} in Figure 3 is taken for illustrate. Before implementation of WS_{21} and WS_{41}, it is need to evaluate mutual trust value. The evaluation process includes many steps of communication. In order to get accurate trust value, direct trust and recommendation trust are combined to evaluate it. The entity with the same functionality, such as WS_{22} and WS_{42}, can be served as recommender. WS_{22} has direct interaction experience with WS_{41} and evaluates the trust value of WS_{41} and recommends it to WS_{21}. As well as WS_{42} can recommends the trust value of

WS_{21} to WS_{41}. When two trust entities satisfy mutual trust value, the two candidates will be chosen for further execution. The details of trust certification between WS_{21} and WS_{41} are illustrated in Figure 5.

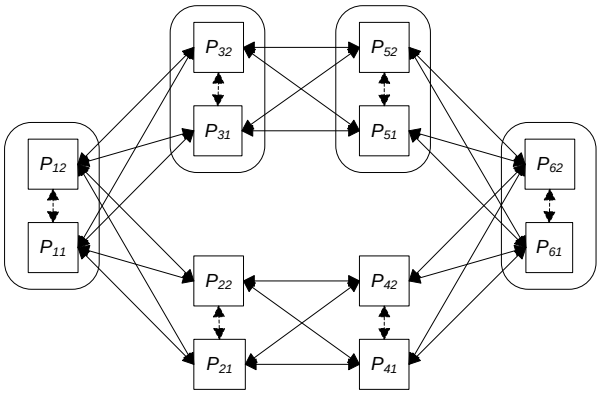

Figure 4. The interaction of trust Web service composition

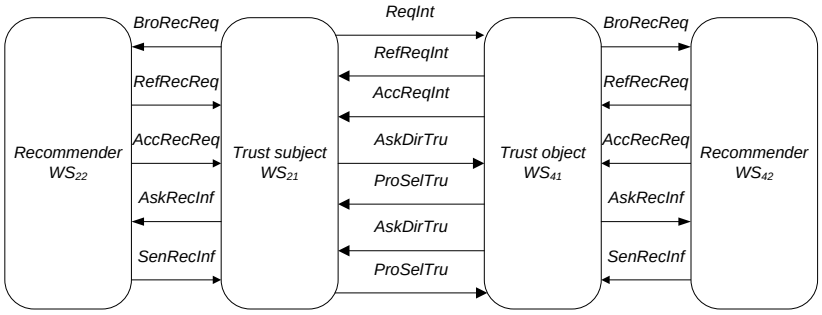

Figure 5. The Details of Trust Authentication

(1) The messages in Figure 5 are interpreted as follows.

- *ReqInt*: trust subject requests trust object to interact.
- *AskDirTru*: trust subject asks the direct trust value of trust object.
- *ProSelTru*: trust subject proposes its own direct trust value to trust object.
- *BroRecReq*: trust subject broadcasts its request for recommendations of trust object.
- *AskRecInf*: trust subject asks recommend trust value of trust object.
- *AccReqInt*: trust object accepts trust subject's request for interaction.
- *RefReqInt*: trust object refuses trust subject's request for interaction.
- *AskDirTru*: trust object asks the direct trust value of trust subject.
- *ProSelTru*: trust object proposes its own direct trust value to trust subject.
- *BroRecReq*: trust object broadcasts its request for recommendations of trust subject.
- *AskRecInf*: trust object asks recommend trust value of trust subject.
- *AccRecReq*: trust recommender accepts request for recommendation.
- *RefRecReq*: trust recommender refuses request for recommendation.
- *SenRecInf*: trust recommender provides recommendation.

(1) Pi-calculus based model of trust certification in Figure 5.

According to Rule 1 and 2, Figure 5 can be converted into the corresponding process graph, as shown in Figure 6. P, Q, R_1 and R_2 represent trust subject WS_{21}, trust object WS_{41}, recommender WS_{22} and WS_{42} respectively. P and Q share the channel x, P and R_1 share the channel y, Q and R_2 share the channel z.

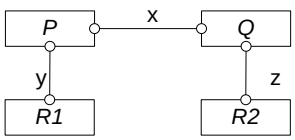

Figure 6. Process graph

Trust subject P contains two concurrent processes P_1 and P_2.

$$P_1(a_1) = \bar{x} < ReqInt > .x(msg1).([msg1 = RefReqInt]P_1(a_1) + [msg1 = AccReqInt]\bar{x} < AskDirTru > .$$
$$x(msg2).[msg2 = ProSelTrs]x(msg3).[msg3 = AskDirTru]\bar{x} < ProSelTru > .P_1(a_1)$$

$$P_2(a_2) = \bar{y} < BroRecReq > .y(\underline{msg}4).([msg4 = RefRecReq]P_2(a_2) + [msg4 = AccRecReq]\bar{y} < AskRecInf > .y(msg5).[msg5 = SenRecInf]P_2(a_2)$$

$$P(a) = P_1(a_1) \mid P_2(a_2)$$

where a_1 = {*x, ReqInt, RefReqInt, AccReqInt, AskDirTru, ProSelTru*}, a_2 = {*AskRecInf, BroRecReq, RefRecReq, AccRecReq, SenReqInf, y*}, $a = a_1 \cup a_2$.

Trust object *Q* contains two concurrent processes Q_1 and Q_2.

$$Q_1(b_1) = x(msg1).[msg1 = ReqInt](\bar{x} < \underline{RefReqInt} > .Q_1(b_1) + \bar{x} < AccReqInt > . x(msg2).[msg2 = AskDirTru]x < ProSelTru > .x < AskDirTru > . x(msg3).[msg3 = ProSelTru]Q_1(b_1))$$

$$Q_2(b_2) = \bar{z} < BroRecReq > .z(msg4).([msg4 = RefRecReq]Q_2(b_2) + [msg4 = AccRecReq] \bar{z} < AskRecInf > .z(msg5).[msg5 = senRecInf]Q_2(b_2))$$

$$Q(b) = Q_1(b_1) \mid Q_2(b_2)$$

where *b*1 = {*x, ReqInt, RefReqInt, AccReqInt, AskDirTru, ProSelTru*}, *b*2 = { *BroRecReq, RefRecReq, AccRecReq, AskRecInf, SenReqInf, z*}, $b = b_1 \cup b_2$.

Trust recommenders *R*1 and *R*2.

$$R_1(c_1) = \underline{y}(msg1).[msg1 = BroRecReq](\bar{y} < RefRecReq > .\underline{R_1}(c_1) + y < AccRecReq > .y(msg2).[msg2 = AskRecInf]y < SenRecInf > .R_1(c_1))$$

where c_1 = {*y, BroRecReq, RefRecReq, AccRecReq, SenReqInf, AskRecInf*}.

$$R_2(c_2) = \underline{z}(msg1).[msg1 = BroRecReq](\bar{z} < RefRecReq > .\underline{R_2}(c_2) + \bar{z} < AccRecReq > .z(msg2).[msg2 = AskRecInf]z < SenRecInf > .R_2(c_2))$$

where c_2 = {*z, BroRecReq, RefRecReq, AccRecReq, SenReqInf, AskRecInf*}.

According to Rule 3, *Q*, R_1 and R_2 can be combined to form a new process, as "Trust service composition", denoted *TSS*, then $TSS(e) = vz(Q(b) \mid R_1(c_1) \mid R_2(c_2))$, $e = b \cup c_1 \cup c_2$.

V. Simulation and Conclusion

The automatic reasoner MWB which is based on SML is chosen to analyze and reason the models above. MWB is an efficient model validation tool set [13]. It is capable of searching for deadlock state, testing for equivalence and checking whether a system has a given logical properties (e.g. safety or liveness).How to judge whether the trust Web service composition is correct or not? The answer is that the system can help trust subject to evaluate the object's trust value, and can meet trust subject's demand on trust quality. Therefore, the trust subject's process is taken as design objective of trust service system, and to analyze whether the trust service system can satisfy trust subject's demand. If the trust service system can meet trust subject's demand, then the trust service system's behavior and trust subject's action are complementary. So, the trust service system's behavior is equivalent to trust subject's dual behavior.

According to process of trust subject, the dual process is defined as follows:

$$DP_1(d_1) = x(msg1).[msg1 = ReqInt](\bar{x} < \underline{RefReqInt} > .DP_1(d_1) + \bar{x} < AccReqInt > . x(msg2).[msg2 = AskDirTru]x < ProSelTru > .x < AskDirTru > . x(msg3).[msg3 = ProSelTru]DP_1(d_1)$$

$$DP_2(d_2) = y(msg1).[msg1 = BroRecReq](\bar{y} < RefRecReq > .DP_2(d_2) + \bar{y} < AccRecReq > . y(msg2).[msg2 = AskRecInf]y < SenRecInf > .DP_2(d_2))$$

$$DP(d) = DP_1(d_1) \mid DP_2(d_2)$$

where d_1 = {*x, ReqInt, RefReqInt, AccReqInt, AskDirTru, ProSelTru*}, d_2 = {*y, BroRecReq, RefRecReq, AccRecReq, SenReqInf, AskRecInf*}, $d = d_1 \cup d_2$.

The channel *z* in trust service system is an internal channel, the actions through this channel can not be observed by trust subject. Both trust service system and dual process have the same observable action set. The simulation results are shown in Figure 7. The trust service system dose not exist deadlocks and circulation, namely system is active. Using step command to track both *TSS* and *DP*, it can be found that although they have different internal structure, their external behaviors are same. Therefore, trust service system can meet trust subject's demand effectively. The behavior of trust Web service composition system can be analyzed and reasoned, thus errors can be found and corrected at design phase, avoiding running time errors. The results show that the formal model on the basis of Pi-calculus is feasible and effective. The future work is to refine evaluation model of trust value and establish trust transfer mechanism for Web service.

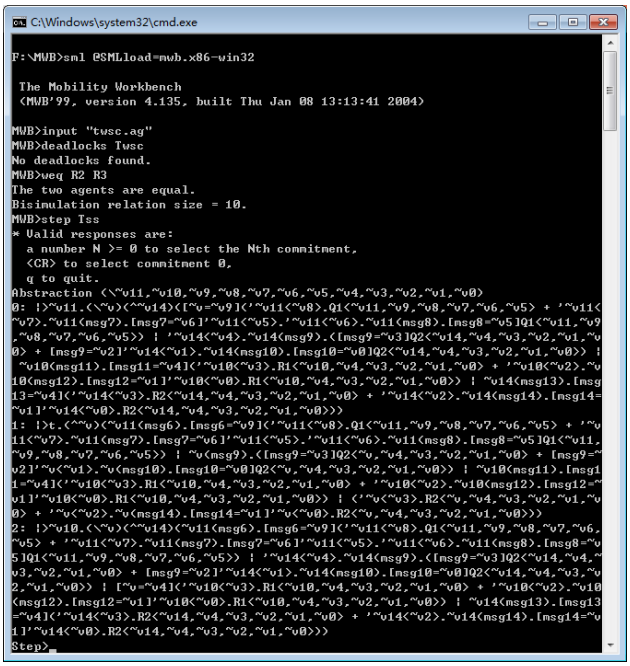

Figure 7. The simulation results

Acknowledgments

The authors would like to thank the anonymous referees for their valuable comments and suggestions of improvements. This work is partly funded by the National Natural Science Foundation of China (Grant No.11204272), the Scientific Project of Zhejiang Provincial Science Technology Department (Grand No.2015C33088) and the State Scholarship Fund of China (File No.201508330728).

References

[1] C.X. Shen, H.G. Zhang, D.G. Feng, et al, Survey of information security, *SCIENCE CHINA Information Sciences*, 50(3), 2007, 273~298.

[2] W. Wang, G.S. Zeng, Bayesian cognitive trust model based self-clustering Algorithm for MANETs, *SCIENCE CHINA Information Sciences*, 53(3), 2010, 494~505.

[3] B.S. Yun, J.W. Yan, M. Liu, Method to optimize Web service composition based on bayes trust model, *Computer Integrated Manufacturing System*, 16(5), 2010, 1104-1111.

[4] B.S. Yun, A trust value computing approach for Web service, *Applied Mechanics and Materials*, 263-266, 2013, 3151-3154.

[5] T. Ayman, K. Ayman, C. Ali, et al, Fuzzy reputation-based trust model, *Applied soft computing*, 11(1), 2011, 345-355.

[6] G. Yin, H.M. Wang, L. Yuan, et al, Construction of internet-based trustworthy software production service system, *Journal of Frontiers of Computer Science and Technology*, 5(10), 2011, 880-890.

[7] L. Roberto, M. Manuel, A Pi-calculus based semantics for WS-BPEL, *The Journal of Logic and Algebraic Programming*, 70(1), 2007, 96-118.

[8] C. Sofiane, B. Faycal, C. Allaoua, A high-level Petri Net based approach for modeling and composition of Web service, *Procedia Computer Science*, 9, 2012, 469-478.

[9] K.G. Hao, X.Q. Guo, X.N. Li, The Pi^{+} calculus – an extension of the Pi calculus for expressing Petri Nets, *Chinese Journal of Computers*, 34(2), 2011, 193-202.

[10] Germano, Walking the web of trust, *the 9th Workshop on Enabling Technologies*, Gaithersburg, MD, 2000, 153-158.

[11] J.C. Fan, K.F. Wu, Introduction to statistical inference, (Beijing: Science Press, 2001)215-251.

[12] S. Davide, W. Davide, The Pi-calculus: A Theory of Mobile Processes,(Cambridge: Cambridge University Press, 2003).

[13] B. Victor, F. Moller, The Mobility Workbench: A tool for the Pi-calculus, The University of Edinburgh, Report number: ECS-LFCS-94-285, 1994.