Web Service Selection Using Quality Criteria and Trust Based Routing Protocol

Mahdi Bazarganigilani

Charles Sturt University, Australia
Mahdi62b@yahoo.com

Abstract

Web Services are applications that perform desired tasks. Such as basic network connectivity to sophisticated compound tasks. Service Composition is the construction of complex services to enable different tasks. Therefore, enabling a rapid and effective composite of services is crucial point in efficiency of composite web services. This paper introduces a new algorithm for effective construction of the primitive services according to their quality criteria. A trust based ranking algorithm is employed to diminish the service with lower qualities.

Keywords: *Composite Service Construction; Trust Based Ranking; Web service Qualities; Path Selection Strategy*

1. Introduction

Web Services are defined applications with URI (Unified Resource Identifier) which interconnects with each other through encoded messages. Such messages use XML (Extensible Markup Language), SOAP (Simple Object Access Protocol), WSDL (Web Service Description Language), UDDI (Universal Description Discovery and Integration) to locate each other. Such technologies are transmissible through Internet protocols [1].

Web Services structure provides the facility to create a distributed service structure. However, the major step is making an efficient composite of primitive services to accomplish the tasks [2].

There is plenty of research on this subject [3, 6]. However, there are many web services with different qualities and efficiencies. Moreover, the users request their needs according to different criteria. Therefore, there should be an effective decision algorithm to satisfy customers' needs. Selecting the web services according to the quality measurements plays a key role in decision-making algorithms [7, 8]. In some methods, the researchers have applied the quality criteria according to the semantic web [9] and in others, they have used basic quality parameters [10]. The second approach determines the qualities criteria more effectively. This paper uses a trust based ranking approach to score or penalize the service from the feedback of every request. In this manner, an effective criterion is employed to select better services while facing similar ones.

Similar works have been proposed by Phan, et. al., [11]. Lai et al proposed a new implementation of SOAP messaging on UDP protocol [12]. It is concluded, the implementations work more effectively on UDP due to the inconsistent nature of the protocol [13]. In the following sections, the composite services and their primitive constituents are described. Second, the trust based routing algorithm and its application to primitive nodes are proposed. Consequently, selection of the most efficient composite services from different

possible combinations is explained. Finally, the evaluation results and comparison with non trust based methods are presented.

2. Web Service Selection Using Quality Criteria

ISO standards [14] define quality, including a number of non-applicable attributes such as cost, response time and availability. The total score of composite service is calculated using the primitive score of each component. This paper focuses on response time and availability of each service. Furthermore, each service is scored by receiving requests. This indicates the service is not in a congestion situation due to high load of the network. A routing algorithm is implemented based on the trust on the primitive services. The total trust is calculated according to the number of responses and requests which the primitive service is receiving.

A Composite service is a compound of a number of primitive services. There are many primitive services based on users' needs. The basic process of a composite service is illustrated with a graph. For each process, there is a node and two starting and ending nodes in the graph. Figure 1 shows a sample graph to get account details of the customers.

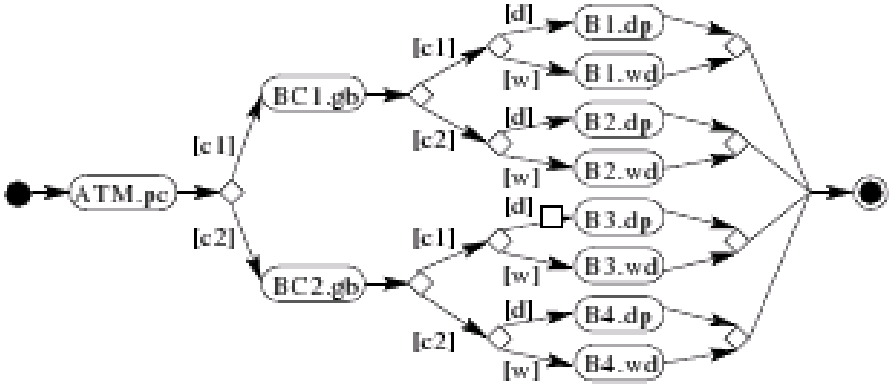

Figure 1. Bank Account Composite Service

In the above example, the primitive service selects the best branch to fetch the information of the particular bank account. In execution time, there are many available web services to accomplish a composite task. Therefore, selecting the best primitive services needs an efficient algorithm to improve total efficiency of the composite service.

According to the above analysis, there are different combinations of tasks for a composite web service. Moreover, composite web services are located in dynamic environments. Thus, the quality of each primitive service changes during different times. By considering the quality parameters mentioned above, the best primitive services based on response times and total path qualities are selected.

Trusted based Routing Algorithm

Mehdipour et al [14] introduced a TCP trust-based routing protocol in faulty environments. The main idea of which was to save a trust score for each node of the composite service. This algorithm saves trust ranks of each node in adjacent nodes. The overall score of any path is computed according to the partial scores of constituent nodes. The trusted-based routing method is completely different from the confidant protocol [15]. This is because, the nodes get the scores based on individual feedbacks and this algorithm gives lower scores to the inefficient services. Therefore, it improves the performance of the overall composite service.

Different modules and strategies are used to update the trust score of every node. For implementing such routing algorithm, the following steps are defined:

- Nodes are initialized.
- Nodes' trust scores are updated.
- The most efficient path. is selected.

In the above steps, a primitive structure is created. This structure maintains the feedback of one node in relation to other nodes. Moreover, it updates nodes' scores and finally selects the best path in the composite service. This structure employs SOAP responses and requests to evaluate the feedback of each node.

In the initialization phase, the trust value of each node is not defined, and the algorithm does not have any information from inefficient services. The algorithm trusts known services more than completely unknown nodes which get the least trust among their adjacent nodes [16]. Consequently, the trust scores are updated based on successful and unsuccessful responses.

3. Trust Updating Strategy

This section discusses the updating function. Trust is an intellectual value. Therefore, it is not fixed in all situations. It changes in different situations. Trust is dependent on other factors such as, previous trust values, maximum and minimum value of the trust and the number of previous positive and negative feedbacks [16].

This study uses the following formula for updating the trust of each primitive service.

$$f_d(ev, tv) = d * tv + (1-d) * ev$$

In the above formula, is the current trust and is the result feedback. In this manner, the new trust depends on both previous gained trusts and the current received feedback. The current received feedback depends on the response time. Moreover, is the constant parameter showing the speed of convergence. The feedback is defined proportional to the response time. Therefore, the less amount of delay is resulted with higher feedback. The output feedback is normalized between 1 and 0 using bellow formula.

$$ev = \begin{cases} 1 & if \quad \text{Re}sponse < t_{\min} \\ \frac{2}{t_{\max} - t_{\min}} * \text{Re}sponse + 1 - \frac{2 * t_{\max}}{t_{\max} - t_{\min}} & if \quad t_{\max} < \text{Re}sponse < t_{\min} \\ 0 & if \quad \text{Re}sponse > t_{\max} \end{cases}$$

Congestion paths and nodes involved in such routes have bigger amount of response time. Such nodes gradually receive less trust. Such negative trusts give higher preferences to alternative paths.

The response time depends on other parameters. Such parameters depend on the current situation of connectivity in the composite network. Malfunctioning node increase the amount of unsuccessful responses. Malfunctioning nodes discard most of requests were received for completing ting the composite service

Considering response time uniquely, is not an effective way. For including the effect of malfunctioning nodes, another parameter is considered for updating the trust of each node. In this manner, receiving each successful request increases the node's trust with 0.7. Figure 2, shows the repetition of different feedbacks with different values of d.

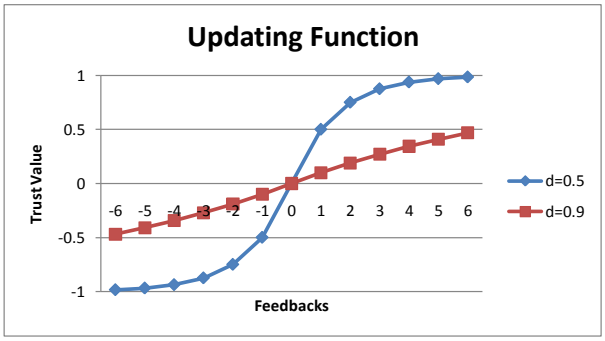

Figure 2. Trust Updating Function

As seen 0.7 is the mean between two values illustrated. This parameter tunes the convergence speed of trust vale to the maximum or minimum trust. Malfunctioning nodes with persistent downtime consecutively receive negative trusts. With values such as 0.5, the final trust converges to minimum trust with faster rates. If the primitive node receives more consecutive positive trusts, it rapidly converges to higher trusts. This value should be tuned based on the overall throuput of the entire composite networks. Nodes should be initialized with the trust resulted with the higher rate of successful reposes. In the experiment section, the best value is obtained for this parameter. As Figure 2, suggests, the values near to 1 results in less convergence and more resistance in updating the trust of primitive nodes. On the other hands, values such 0.5, reflects the dynamic changing of trusts more effectively. While consecutive similar trusts result in faster convergence to higher or lower trust, the more similar trusts will not update trust much more. This is very effective in congestion period times. While congestion can occur on an instance of the time, the trust is updated quickly to reflect the congestion. The consecutive negative trusts will not change the trust much further. By alleviation of congestion, the node receives more positive trusts and it quickly converges to higher trusts.

4. Path Selection Strategy

Path selection strategy includes selecting the best path with maximum trust scores. The path involved with inefficient and slow services do not get higher trusts. This is crucial for composite service to select the alternative primitive services. This section introduces trust evaluation of possible paths. As described in previous section, one possible service can have a trust value of 1 at most. It denotes its efficiency in composite web service. The more inefficient primitive services are resulted in lowers trust composite network combination [14].

This paper makes use of deviation of each service to evaluate the effectiveness. Therefore, more deviations from 1 indicate more inefficient combinations. The most insecure path is the path with all trust value, -1. In this case, the standard deviation is 2. To prioritize the lowest number of primitive services in one combination, the output trust divided by the number of nodes [12].

$$T(Route) = \frac{2n - \sqrt{n\sum_{i=1}^{n}\left(T(y_i)-1\right)^2}}{n}$$

In above formula, $T(Route)$ is the total trust rank and $T(y_i)$ is the rank of node y_i. Table 1, illustrates an example.

Table 1. Trust Path Selection Strutegy

Path	Trust Values				T(Route)
Path 1	-0.3	-0.3	-0.3	-0.3	0.17
Path 2	-0.35	0.9	0.9	1.0	0.33

This algorithm gives better score to path 2. However, by considering minimum strategy, path 1 was selected. While, the most trustless node exists in path 2, the algorithm gives higher score to the path with higher average trust. Another example is shown in Table 2. Proposed selection strategy represents the trust of path according to the overall score of encompassed primitive nodes. In this manner, the path higher overall trust with a persistent malfunctioning node can have predominance to another path with many malfunctioning and congestion points.

Table 2. Trust Path Selection Strutegy

Path	Trust Values				T(Route)
Path 1	0.3	0.1	-0.2	-	0.17
Path 2	0.4	0.1	0.1	0.2	0.33

This algorithm gives more trust to Path 1. This path consists of less numbers of nodes, On the other hand, employing average approach can not distinguish the path with higher trust. Proposed selection strategy defines the paths with less numbers of nodes as higher trust with equal average trust for consisting primitive services.

5. Evaluation Results

SOAP is a text-based protocol for Web services, but it has a high overhead and redundancy. Existing Web services frequently rely on HTTP. While the HTTP protocol provides a number of advantages, it was developed for trusted wired networks with high bandwidth, low latency and faulty rate transmissions. Due to the variability of integrity among channels, these assumptions are inappropriate in low trust routing environments. Moreover, there can be malicious services and major malfunctioning nodes. To overcome the drawbacks, Phan, et. al., [13] introduced SOAP over UDP. They concluded SOAP over UDP works much more efficiently than TCP. However, this is correct while the environment lacks efficiency in routing paths. If the paths have some major constant malfunctioning points, UDP is not a reliable protocol. It exacerbates the ineffectiveness due to lack of any reliable mechanism for message exchange. Therefore, there is no possibility to discover faulty nodes. This paper shows various experiments both on TCP and UDP based on the proposed trusted routing algorithm.

Simulations were implemented by 50 primitive services which routed the requests to some central services. All modules and algorithms were implemented with the .NET framework. Inefficient services did not forward any request. Therefore, no response was received and the

trust score remained -1. This paper defines another factor named throughput as the amount of received responses divided by total number of requests in source nodes.

0,10,20,30 and 40 percent of primitive services were considered inefficient. The trust scores were initialized with different values for unknown services. Figure 3, shows the throughput for different initialization values.

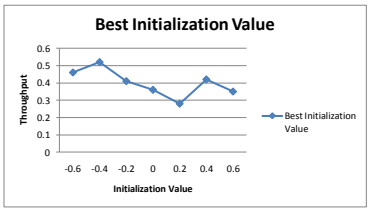

Figure 3. Throuputs for Different Initilization Values

As seen, the best value was resulted from $d = -0.4$. This paper used this initialization trust value for the remaining g experiments.

In experiment II, 0 to 40 percent of services were selected as inefficient. The experiment was conducted to obtain total throughput. Figure 4, shows the throughput in trust based routing implementation.

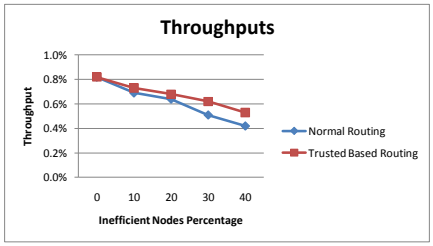

Figure 4. Throughput Comparison beween Trusted Absed Routing and Normal Protocol

As concluded, trust based algorithm outperforms the normal routing in the existence of inefficient nodes. This is obviously resulted in implementing trust value for each primitive service. Such trusts represent the trustworthy of each primitive node. Nodes with high loads of unsuccessful responses receive the negative trusts and are not considered for composite services.

In experiment III, the simulations were conducted over UDP and TCP.

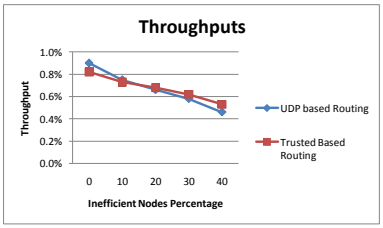

Figure 5. Throughput Comparison Between UDP and TCP Trusted Based Routing Method

As illustrated in the Figure 5, when there were few malfunctioning nodes, the UDP based protocol outperformed much effectively. However, by increasing the number of faulty nodes, TCP trust-based routing protocol showed more reliable throughputs. This experiment indicates that UDP based protocol is not able to distinguish faulty nodes in an environment with constant malfunctioning services. This property is more tangible when the faulty nodes are persistent. Such nodes consequently receive the constant negative trusts. These nodes remain with negative trusts in the period of downtime or congestions. Consequently, such primitive services result in les score for the encompassing path. In such situation, the alternative paths with higher trusts, which do not include such persistent faulty nodes, are selected. Whole UDP protocol does not handle the correctness delivery of requests. It is supposed to be an effective protocol for such environments with existence of faulty nodes. This experiment shoes the incapability of such protocol to distinguish the sinks node with constant congestion and delay times. While UDP dos not implement any mechanism to avoid such nodes, the proposed protocol suggests a method to avoid such sink nodes in period of down times. Obviously, the throughput of faulty environment is improved by employing trust-based protocol. This difference is more tangible in persistent malfunctioning service due to the congestion traffics.

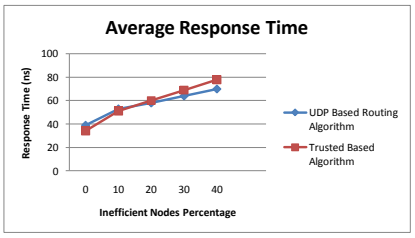

Figure 6. Average Response Time Comparison between UDP and TCP Trusted Based Routing

While it was supposed, UDP outperformed in a malfunctioning environment [13], the results indicate the effectiveness of the proposed method. This is because by increasing the number of faulty nodes, the overhead time of TCP was compensated by TCP trust based routing protocol.

Trust based routing protocol efficiently computes the trust of each path. The trust of each path is calculated based on each node. The overall trust is not solely dependent to each node. It is dependent to overall trust of path. Paths with less numbers of nodes will result in higher trusts. While UDP does not support the sequencing of the HTTP responses and request. It was supposed to perform much effectively in faulty environments. However, in persistent faulty environments, which there can be nodes affected by malicious programs, the trust-based routing protocol performs effectively. It efficiently distinguishes the path with the malfunctioning nodes. Such persistent sinks in the network causes the lower trust in the intersection paths. For consecutive delivery of the requests the trust of such paths are decreased and the protocol suggests alternative path with higher value of overall trust.

Another advantage of trust-based routing protocol refers back to the dynamic property of such protocol. While, there can be many congestion node due to high demand. The congestion results in long delays in HTTP responses. Such long responses result in lower score to each primitive service. Each primitive service with high load of unsuccessful responses receives negative trusts. By decreasing the number of request, the response time decreases and the rate of successful responses increase. Consequently, the trust of primitive service is improved. This property is very common in HTTP protocol and communication networks. Each primitive service has an updating trust, which reflects the trust of each node in the current time. It can be updated in forwarding times. It can be decreased due to the congest traffics, malicious attacks or downtime periods. It can be improved due to the less request and the less response times.

6. Conclusions

This paper introduced a new trust based algorithm for selection of the most efficient primitive services. This method assigned the trust based ranking values dynamically. Therefore, the number of correct responses and requests reached a primitive service was employed to rank the trustworthiness of each node. This paper employed the number of responses and requests of each node. These request and responses created the trust of each node. The overall path trust is calculated using such primitive trust values. Trusts were updated according successful response received. More rapid responses are resulted in achieving higher trusts.

Various experiments were conducted. Results showed that in a faulty dominated environment, proposed protocol performs more effectively than previous methods. This is due to employing the trust value for each node and subsequently for the entire path. Such paths represented the higher possible trusts in comparison to any other alternative paths. The composite service consisted of more effective primitive services. This results in higher throughput and success rate of received requests and responses. The trusts were updated according to the dynamic situation of the network. The congestion or malfunctioning nodes can be alleviated which result in trust improvement of primitive nodes. This enhanced the overall trust of composite service. While it was supposed UPD protocol would perform much better in faulty environments due to neglectance to correctness delivery of the messages, the proposed method suggested a trust based routing protocol to clearly define the trust of primitive node for engaging in composite services. Such trusts were updated to accompany with dynamic situation of the connections. Such trusts reflect the most effective way of consisting of a composite primitive services. Such trusts were employed to select best primitive nodes to be selected for composite service. Such selection resulted in higher throughput and success rate of responses. Moreover, response times increased due to the usage of trust values for consistent faulty services.

References

[1] W3C.WebServicesArchitecture, http://www.w3.org/TR/2004/NOTE-ws-arch- 0040211/, **(2004)**.
[2] F. Curbera, R. Khalaf and N. Mukhi, "The Next Step in Web Services", Communication of the ACM, vol. 6, no. 10, **(2003)**, pp. 29-34.
[3] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam and Q. Z. Sheng, "Quality Driven Web Services Composition", Proc. 12th Int'l Conf. World Wide Web (WWW), **(2003)**.
[4] G. Canfora, M. D. Penta, R. Esposito and M. L. Villani, "A lightweight approach for QoS–aware service composition", Proceedings of the 2nd International Conference on Service Oriented Computing, (ICSOC'04), New York, USA, **(2004)**, pp. 36–47.
[5] N. Milanovic and M. Malek, "Current Solutions for Web Service Composition", IEEE Internet Computing, **(2004)**, pp. 51-59.
[6] B. Orriens, J. Yang and M. P. Papazoglou, "Model Driven Service Composition", In the First International Conference on Service Oriented Computing (ICSOC'03), **(2003)**.
[7] D. A. Menascé, " QoS Issues in Web Services", IEEE Internet Computing, vol. 6, no. 6, **(2002)**, pp. 72–75.
[8] D. A. Menascé, "Composing Web Services: A QoS View", IEEE Internet Computing, **(2004)**, pp. 88-90.
[9] M. Tian, A. Gramm, H. Ritter and J. Schiller, "Efficient Selection and Monitoring of QoSAware Web Services with the WS-QoS Framework", IEEE/WIC/ACM International Conference on Web Intelligence, (WI'04), **(2004)**.
[10] C. H. Zhou, L. T. Chia and B. S. Lee, "DAML-QoS Ontology for Web Services", IEEE International Conference on Web Services (ICWS'04), **(2004)**.
[11] K. A. Phan, Z. Tari and P. Bertok, "Enhanced SOAP Performance For Mobile Applications", In Proceedings of the 21st Annual ACM Symposium on Applied Computing, Dijon, France, **(2006)** April, pp. 1139–1144.
[12] K. Lai, K. A. Phan and Z. Tari, "Efficient SOAP Binding for Mobile Web Services", In Proceedings of the 30th Annual IEEE Conference on Local Computer Networks, Sydney, Australia, **(2005)** November, pp. 218–225.
[13] K. A. Phan, "Enhanced SOAP Performance for Low Bandwidth Environments", master thesis, School of Computer Science and Information Technology , RMIT University, Victoria, Australia, **(2007)**.
[14] E. Mehdipour, A. Movaghar and A. M. Rahmani, "A New Trusted Based DSR Protocol Routing", 13th Int'l CSI Computer Conference (CSICC'2008), Sharif University of Technology, **(2008)** March 9-11, Kish Island, Persian Gulf, Iran.
[15] ISO 8402, Quality Vocabulary.
[16] F. Stajano and R. Anderson, "The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks", In Proceedings of the 7th International Workshop on Security Protocols.
[17] C. M. Jonker and J. Treur, "Formal Analysis of Models for the Dynamics of Trust Based on Experiences", Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World: MultiAgent System Engineering, **(1999)** June 30-July 02, pp. 221-231.