Modeling and Analysis of Contract Net Protocol

Fu-Shiung Hsieh

Overseas Chinese Institute of Technology
407 Taichung, Taiwan, R.O.C.
shie1210@ocit.edu.tw

Abstract. Contract net protocol can be implemented by exploiting the Web Service technologies. However, the lack of a process model in contract net protocol to capture the interactions between the bidder and manager agents makes it difficult to analyze the feasibility of the resulting contracts. We proposed a framework to execute contract net protocol based on Web Services technologies and a Petri net model to analyze the contract net negotiation results.

1 Introduction

Contract net protocol [1] is a negotiation and task distribution mechanism to optimize the performance in multi-agent systems (MAS) [2]. Contract net protocol can be implemented by exploiting the Web service discovery, invocation, selection and execution mechanism. In this paper, we focus on modeling and analysis of contract net protocol that can be implemented based on Web Services technologies [3]. Given a set of tasks, a set of agents, a cost function, contract net protocol finds the globally optimal task allocation based on distributed algorithms. In contract net protocol, there are two types of agents: manager and bidder. Four stages are involved to establish a contract between a manager and the best bidders. To apply Web Services technologies to execute contract net protocol, each manager is mapped to a Web services requester whereas each bidder is mapped to a Web services provider. Each bidder (which is a service provider) publishes the Web services by registering through a Web services broker. A manager (which is a service requester) looks up the registries of a Web service broker to discover qualified Web services provided by the potential bidders. Fig. 1 (a) illustrates our concept. Once the required qualified Web services have been found, the manager executes the contract net protocol to award the contract to the best bidder as shown in Fig. 1(b)~(e). By combining the contract net protocol with Web services technologies, it is possible to automate negotiation process.

A major drawback of the original contract net protocol is the lack of a formal model to capture the interactions between bidders and managers as well as the negotiation results. This makes it difficult to analyze the feasibility of the contract net negotiation results. For example, in real world, several contracts may need to be established between a company and his business partners to achieve a certain business objective. The lack of a model makes it difficult to efficiently evaluate the feasibility of the contracts. In this paper, we will propose a Petri net [4] model to model the

N. Koch, P. Fraternali, and M. Wirsing (Eds.): ICWE 2004, LNCS 3140, pp. 142–146, 2004.

interactions between the bidders and the managers in contract net protocol and facilitate the feasibility analysis of the contract net negotiation results.

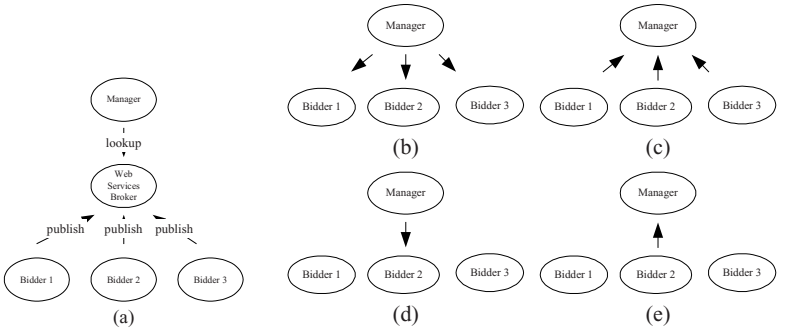

Fig. 1. Execution of contract net protocol based on Web Services technologies

2 Petri Net Model for Evaluating Contracts

To model the interactions between the bidders and the managers in contract net protocol, we adopt a bottom-up approach using Petri nets, which can be broken down into three steps: (1) modeling of bidders' proposals, (2) modeling the workflow of tasks and (3) modeling the overall system.We assume there is a set B of bidders in the system and each bidder $b \in B$ has a set R_b of type-b resources available to bid for several contracts. We use P_b to denote the states of type-b resources and place p_{b0} to denote the idle state of type-b resources. Allocation and de-allocation of the resources by bidder b are represented by a circuit that starts and ends with place p_{b0}. The Petri net G_b to describe a bidder's proposal is defined as follows.

Definition 2.1: The Petri net $G_b = (P_b, T_b, I_b, O_b, m_{b0}, F_b)$ of a bidder $b's$ proposal is a strongly connected state machine (SCSM) with any two circuits in G_b having only one common place p_{b0} but having no common transitions.

The Petri net $G_b = (P_b, T_b, I_b, O_b, m_{b0}, F_b)$ represents a proposal submitted by the bidder $b \in B$. Fig. 2 illustrates seven bidders' proposals using Petri nets. We assume that there is a set J of different types of tasks in the system. A type-j task, $j \in J$, is modeled by a Petri net to executing a sequence of operations.

Definition 2.2: The Petri net $G_j = (P_j^w, T_j^w, I_j^w, O_j^w, m_{j0}^w, F_j^w)$ of the workflow of a type-j task, $j \in J$, is a strongly connected state machine (SCSM). Every circuit in G_j contains the idle state p_{j0}, the starting transition t_j^r, the final transition t_j^f, the initial state place p_{ji} and the final state place p_{jf}, with $p_{j0}^{\bullet} = \{t_j^r\}$, ${}^{\bullet}p_{j0} = \{t_j^f\}$, $t_j^{r\bullet} = \{p_{ji}\}$ and ${}^{\bullet}t_j^f = \{p_{jf}\}$. As each transition represents a distinct operation in a task, we assume $T_j \cap T_k = \Phi$ for $j \neq k$.

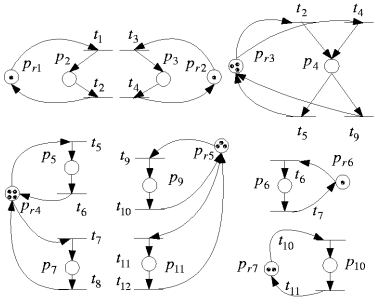

Fig. 2. Bidders' Proposals

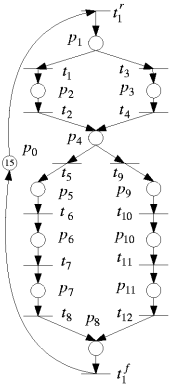

Fig. 3. A Task Workflow

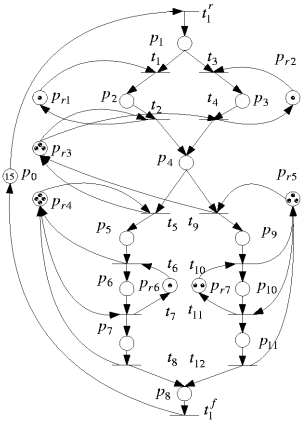

Fig. 4. Petri net Model

Fig. 3 illustrates the Petri net model of a task workflow. Place p_1 represents the initial state place while place p_8 represents the final state place. All the other places in this Petri net represent the states of the task. Each directed path from place p_1 to p_8 represents an execution sequence for the task. The Petri net models for the bidders' proposals and the task workflow are merged to form a Petri net $N_j(m_{j0}) = \|_{b\in B} G_b \,\|\, G_j = (P_j, T_j, I_j, O_j, m_{j0}, F_j)$ to model the interactions between resources and a task, where $\|$ is an operator to merge a number of different Petri net models with common transitions, places and/or arcs.. The Petri net model in Fig. 4 is obtained by merging the proposals in Fig.2 with the task in Fig. 3.

3 Feasible Condition to Award Contracts

The Petri net N_j provides a model for a manager to award contracts. To characterize the condition to complete a type- j task, we define a feasible execution sequence as follows.

Definition 3.1: Given a Petri net N_j with initial marking m_{j0}, a feasible execution sequence $s = t_1 t_2 t_3 ... t_{|s|}$ with $t_1 = t_j^r$ and $t_{|s|} = t_j^f$ is a firing sequence such that firing s brings a token from the initial state place p_{ji} to the final state place p_{jf}.

As $N_j = \|_{b\in B} G_b \,\|\, G_j$ and G_j is a connected state machine, there may be more than one execution sequence for a type- j task. However, an arbitrary execution sequence might not be a feasible one. For example, in Fig. 5, $t_1 t_2 t_9 t_{10} t_{11} t_{12}$ is not a feasible execution sequence due to the lack of a type-7 resource in place p_{r7}. In Fig. 5,

$t_1t_2t_5t_6t_7t_8$ and $t_3t_4t_5t_6t_7t_8$ are feasible execution sequences. A feasible execution sequence can be characterized based on the minimal resource requirement (MRR) concept.

Definition 3.2: The MRR of an execution sequence $s = t_1t_2t_3....t_{|s|}$ is denoted by a marking m_s^* of N_j with $m_s^*(p) = \begin{cases} N_s^*(b) & \text{if } p = p_{b0} \text{ for some } b \in B \\ 0 & \text{otherwise} \end{cases}$,

where $N_s^* = R_{t_1} \oplus R_{t_2} \oplus R_{t_3} \oplus \ldots \oplus R_{t_{|s|}}$, a vector in $Z^{|B|}$, denotes the set of resources required to fire s, R_t, a vector in $Z^{|B|}$, denotes the resource requirement to fire t only, and $\oplus$ is an operator that takes the larger of two vectors element by element.

Existence of a feasible execution sequence for $N_j(m_{j0})$ is as follows.

Property 3.1: Given $N_j(m_{j0})$, s is a feasible execution sequence if and only if $m_{j0} \geq m_s^*$.

Remark that N_j is constructed by merging G_j with G_b for all $b \in B$. Each execution sequence of N_j must be an execution sequence of G_j. Each execution sequence $s = t_1t_2t_3...t_{|s|}$ of G_j corresponds to a directed path γ_s from t_j^r to t_j^f in G_j. Therefore, to find a feasible execution sequence for a type- j task, it is sufficient to find a directed path γ_s from t_j^r to t_j^f with $m_{j0} \geq m_s^*$.

To award the contracts for a type- j task, we find the minimum cost feasible execution sequence for a Petri net $N_j(m_{j0})$. The problem is formulated as follows. Let c_t denote the cost to fire a transition t, where $t \in T_j$. The cost to fire the sequence $s = t_1t_2t_3...t_{|s|}$ of transitions is then $c(s) = \sum_{t \in s} c_t$. Let S_j denote the set of all feasible execution sequences of $N_j(m_{j0})$. The optimization problem can be formulated as the problem to find $\min_{s \in S_j} c(s)$. The problem to find $\min_{s \in S_j} c(s)$ is equivalent to solve $\min_{\gamma \in \Gamma_j^{feasible}} c(\gamma)$, where $\Gamma_j^{feasible} = \{ \gamma \mid \gamma \text{ is a directed path from } t_j^r \text{ to } t_j^f \text{ and } m_{j0} \geq m_\gamma^* \}$ denotes the set of all feasible directed paths in N_j.

Definition 3.3: Let $T_j^{feasible} = \{ t \mid t \in T_j, R_t \leq m_{j0} \}$ and $T_j^{\inf\ easible} = T_j \setminus T_j^{feasible}$ be the set of transitions that cannot be fired due to insufficient resource tokens in m_{j0}.

The problem $\min_{\gamma \in \Gamma_j^{feasible}} c(\gamma)$ can be converted to a shortest path problem by constructing a weighted directed graph $\tilde{G}_j$ based on G_j and $T_j^{\inf\ easible}$, where $\tilde{G}_j$ can be constructed by

removing p_{j0} and the set $T_j^{\inf\ easible} \cup \{ t_j^r , t_j^f \}$ of transitions from G_j, and then converting each transition in the resulting graph into a directed arc and assigning a weight to it. The weight w_t of a transition t is c_t for each $t \in T_j^{feasible}$. The shortest path problem described by $\widetilde{G}_j$ can be solved by applying the Dijkstra's algorithm with polynomial complexity. Fig. 6 illustrates the Petri net $\widetilde{G}_j$ obtained based on G_j in Fig. 3 and $T_j^{\inf\ easible} = \{ t_{10} , t_{11} \}$. Existence of a shortest path depends on $\widetilde{G}_j$ and marking m_{j0}.

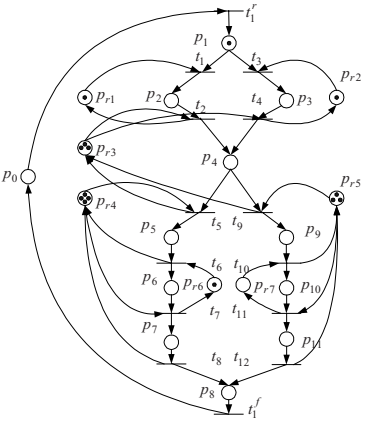

Fig. 5. An N_j with Infeasible Execution Sequence

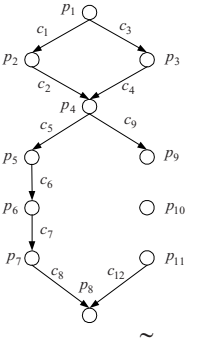

Fig. 6. $\widetilde{G}_j$

4 Conclusion

We focused on modeling and analysis of contract net. We identified a feasible condition to award contracts and formulated an problem to find a minimal cost feasible execution sequence for a task based on Petri net. The optimization problem can be converted to a shortest path problem, which can be solved efficiently.

References

1. R.G. Smith, "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver", *IEEE Trans. On Computers* Vol. 29, pp. 1104-1113, 1980.
2. Jacques Ferber, "Multi-Agent Systems, An Introduction to Distributed Artificial Intelligence," Addison Wesley, 1999.
3. S. McIlraith, T.C. Son, H. Zeng: Semantic Web Services, IEEE Intelligent Systems, Special Issue on the Semantic Web, 16 (2) (2001) 46-53.
4. Tadao Murata, "Petri Nets: Properties, Analysis and Applications," *Proceedings of the IEEE*, vol. 77, No. 4, April 1989.