The manufacturing resource optimization problem of federal collaborative development based on Markov Decision Process

Qiang Qin$^{1, 4}$, Jianxin Yang$^1$, Jun Ji$^1$, Wenjun Liu$^{1, 2}$, Yi-ming Yang$^1$ and Wei Wei$^3$

$^1$ Information Center of China North Industries Group corp, Beijing 100089, China
$^2$ School of Computer, Beijing Institute of Technology, Beijing 100081, China
$^3$ School of Mechanical Engineering and Automation, Beihang University, Beijing, China, 100191
$^4$ E-mail: 8226539@163.com

Abstract. To solve the problem of optimal allocation of cloud manufacturing resources considering of the uncertainty during the product development process, a Markov decision model for the development of the capability unit assignment is proposed. And the cross-entropy method is adopted to obtain the optimal allocation strategy of the developed capability unit by the learning strategy. The simulation results show that the optimal strategy is better than the stochastic allocation strategy, which can reduce the cost of product development and improve the efficiency of product development.

1. Introduction

The idea of Cloud manufacturing is coming out of the extensive use of cloud computing technology, the Internet of things and the service-oriented architecture. Cloud manufacturing is a service-oriented, efficient and knowledge-based new model of intelligent manufacturing [1-3]. It is better to respond the manufacturing demands of the market by integrating the social manufacturing resources and capabilities. It makes the distributed resources can be shared or be collaborative work together. The core that achieves this manufacture mode is to build suitable resource allocation strategy. It is an urgent problem to be solved in the implementation process of cloud manufacturing.

Recently some research results have been carried out by some scholars focus on optimization of resource under the manufacturing environment. According to the characteristics and problem of manufacturing resource scheduling in cloud manufacturing environment, Literature [4] built a multi-objective optimization mathematical model. Considering time, cost, quality and ability a manufacturing resource scheduling method was presented based on genetic ant colony algorithm. To explore more effective optimization configuration algorithm for collaborative manufacturing resources, Literature [5] proposed a new mathematic model based on Adaptive Ant Colony Algorithm (AACA). Literature [6] established a resource optimization model with the least cost, the least time and the most quality based on formal description of resource allocation problem in cloud manufacturing was established by considering the influence of material flow and information flow on time and cost. To optimize the allocation of cloud service resources effectively and to reduce the cost of manufacturing services, by considering the influence of material flow and information flow on time and cost, a resource optimization model with the least cost, the least time and the most quality based on formal description of resource allocation problem in cloud manufacturing was established. Max Inherit
Optimization (MIO) approach was used to solve the model. An optimization objective system included eight variables which were T (service Time), Q (service Quality), C (service Cost), A (Availability), Co (Compositionality), Tr (Trust), Ma (Maintainability), Su (Sustainability) was put forward in Literature [7]. From this viewpoint, an optimization model of cloud manufacturing services resource combination for new product development was established, and grey relational analysis method was adopted to analyze the solving process of model. A flexible resource-constrained resource leveling project scheduling problem was proposed to resolve highly-effective utilization of flexible resources for One-of-a-Kind Production (OKP) in Literature [8]. The mathematical model was constructed. To resolve the model, a Path Relinking (PR) algorithm which was based on an Adaptive Serial Schedule Generation Scheme (ASSGS) and a Maximum network flow-based Flexible Resource Assignment Model (MFRAM) was presented. With central management of center console, Service Composition Optimal Selection (SCOS) and Optimal Allocation of Computing Resources (OACR) are two critical steps for implementing high flexible and agile service provision and resource sharing among sub-enterprises and partner-enterprises under the key technologies of virtualization. Two steps decision-making are inefficient and cumbersome. To overcome this deficiency, a method named Dual Scheduling of Cloud Services and Computing Resources (DS-CSCR) was presented in Literature [9].

The above research results optimize the allocation of resources according to variety situation. But they all ignore the enterprise development ability, an important factor under the cloud manufacturing environment. The development of the enterprise capability characteristic determines the task of the enterprise and the uncertainty of task completion in the process of the development. Under the constraints, the limitations of the above results are revealed.

In this paper, considering the feature of cloud manufacturing environment to develop ability to under the influence of manufacturing resource selection, we established a markov decision model of cloud manufacturing resources optimization allocation. To obtain the optimal resource allocation strategy, a fast learning method of optimal allocation of resources is presented based on cross entropy method. Finally, the application of the proposed approach in simulation indicates its effectiveness.

2. The manufacturing resource optimization problem of federal collaborative development

Suppose that there is a complex product development task, which can be decomposed into several subtasks according to performance requirements, structural and accuracy requirements. The complex product development task F can be represented as $F = \{f_i \mid i = 1, 2, \ldots, n\}$, where n means the number of subtasks.

Development capability unit: The capability of an organization to get the special task done is called development capability unit. The development capability unit could be described as $c_{ij} = \{lov(c_{ij}), f_{ij}, j\}$, where $lov(c_{ij})$ means the level of the organization j to complete the task $f_{ij}$. It reflects the expected level of completion of the development task for organization. The higher the level, the higher the success rate of accomplishment the task.

Development capability pool: The development capability pool is a container which is used to manage a certain type capability of development. For the task $i$, its development capability pool is $CP(i) = \{c_{i1}, c_{i2}, \ldots, c_{ig}\}$.

Combination of development capability pool: It is a set of development capability units which meet the needs of a complex product development task. The combination of development capability pool of a complex task which contains $i$ tasks is $CCP = \{CP(1), CP(2), \ldots, CP(1)\}^*$.

In the whole process of development capability unit assignment for a complex product development task, the purpose is to assign a development capability unit for each subtask. Let $S_t = i$ represents that there are $i$ subtasks need to assign at time $t$, then the state set $S = \{S_t, t \geq 0\} = \{0, 1, \ldots, i\}$. An action can be selection of an input value from corresponding
development capability pool. Different selections have different development capability. All possible actions constitute the action space \( A \). At each decision time instance \( t(t \geq 0) \), the current state \( S_t \) and the action to be applied will influence the next state \( S_{t+1} \). The development of task and the capability unit assigning strategy make up a Markov decision process as shown in figure 1.

There are some assumptions as follows:

1. \( s_0 = i \) is the initial state, and \( s_f = 0 \) is absorbing state, it is target state.
2. An action taken at one decision time meets with at most one subtask.
3. At time \( t \) if capability unit \( c_j^i \) is applied in state \( s_t = i \). Then a subtask is finished with probability \( \theta_j \) which making the state \( s_{t+1} = i-1 \) and not finished with probability with \( 1 - \theta_j \) which remaining the state unchanged. Obviously \( \theta_j \) is related with the level of organization \( j \) to complete the task \( i \).
4. If a subtask is not finished, then the number of remaining subtasks remains unchanged.
5. At time \( t \) capability unit \( c_j^i \) incurs a cost of \( C_s(c_j^i(j)) \) where \( s_t \neq 0 \), no matter whether or not it finishes a subtask. The cost is 0 in state \( s_t = 0 \).

Following the above assumptions, the process of development capability unit assignment can be treated as a Markov decision process which is defined by a tuple \( \{T, S, A, P, C\} \), where \( T = \{0, 1, 2, \ldots\} \) is decision time, \( S = \{S_t, t \geq 0\} = \{0, 1, \ldots, i\} \) is state space, \( A = \{CP(1), CP(2), \ldots, CP(i)\} \) is action space, \( P \) is transition probability matrix, \( C : S \times A \rightarrow C \) is set of development cost. From time 0 to \( t \) the Markov decision process of development capability unit assignment history is \( H_t = \{S_0, c_0^1, S_1, c_1^j, \ldots, S_{t-1}, c_{t-1}^i, S_t\} \).

Denote \( \pi \) is a rule that determines for history \( H_t \) the probability distribution of the decision maker’s actions at time \( t \). Let \( \gamma \) is the first time which the state \( S_t = 0 \) is reached the minimal number the capability unit assignment. In rule of \( \pi \) and history of \( H_t \), the sequence of development capability unit is \( X = (c_1^0, c_1^1, \ldots, c_{nj}^{-1}) \). Denote \( Z(X) \) is the excepted cost of development task,

\[
Z(X) = E_\pi \left[ \sum_{t=0}^{\gamma} C_{s_t}(c_t^j) \right] \tag{1}
\]

Where \( E_\pi \) refer to the excepted cost of development task by the strategy \( \pi \).

Our problem is how to seek out the strategy \( \pi^* \) which minimizes the excepted cost of development, that is

\[
\min Z(X) = E_{\pi^*} \left[ \sum_{t=0}^{\gamma} C_{s_t}(c_t^j(k)) \right] \tag{2}
\]
A learning strategy based Cross-Entropy method

Rubinstein [10] introduces the Cross entropy method to estimate the probability of rare events in complex random networks. Though the transformation of Cross entropy, it is solved the probability estimation of rare event. It is an adaptive algorithm that minimizes the estimated variance. Researchers have successfully applied this method to solve the combinatorial optimization problems [11, 12]. Some recent successful applications [13, 14] prove that the cross entropy method is a practical tool for solving the NP-hard problem.

In this section, we will present the Cross-Entropy (CE) method to solve the above problem. The CE method involves an iterative procedure where each iteration can be broken down into two phases:

1. Generate a random sample according to a specified mechanism.
2. Update the parameters of the random mechanism based on the data to produce a “better” sample in the next iteration.

3.1. The Cross-Entropy method

In order to apply the CE method we need to specify (a) how to allocate development capability unit from Development capability pool and how to update the parameters at each iteration. The easiest way is to relate (2) to an equivalent maximization problem.

Denote \( P = (p_{ij}) \) be a \( i \times j \) matrix. We assign capability unit according the state with probability \( p_{ij} \). \( X = (c_1, c_2, \ldots, c_j) \) be a random vector taking values in action space A.

Define a collection of indicator functions \( I_{X \leq \gamma} \) and a family of discrete probability density functions (pdf) \( f(x, v) \) which parameterized by a real-valued parameter (vector) \( v \) on A. For the optimal strategy \( \pi^* \), denote \( f(x, \pi^*) \). We associate (2) with the problem of estimating the number

\[
I(\gamma) = P_u(Z(x) \leq \gamma) = \int I_{Z(x) \leq \gamma} f(x, \pi^*) \, dx = E_u I_{Z(x) \leq \gamma}
\]

Consider the event “cost is low” to be the rare event of interest. To estimate this event, the CE method generates a sequence of tuples, which converge to a small neighborhood of the optimal tuple. We use the importance sampling method to estimate (3)

\[
l(\gamma) = \frac{1}{N} \sum_{i=1}^{N} I_{Z(x^{(i)}) \leq \gamma} \frac{f(x^{(i)}, u)}{g(x^{(i)})}
\]

where \( g(x) \) is important sampling density function and \( x^{(i)} \) is generated by \( g(x) \).

In order to obtain the best important sampling density function

\[
g^*(x) = \frac{I_{Z(x^{(i)}) \leq \gamma} f(x, u)}{g(x^{(i)})}
\]

We need to change the parameter \( V \) by selection probability density function from \( \{f(x, v), v \in V\} \) and minimize the cross entropy between \( f(x, v) \) and \( g^*(x) \). The cross entropy is

\[
KL[g, f] = \int g(x) \log \frac{g(x)}{f(x)} \, dx
\]

which is also

\[
\min_v KL[g^*(x), f(x, v)] = \max_v E_u I_{Z(x) \leq \gamma} \log f(x, v)
\]

Because the sequence \( X = (c_1, f(i, k), c_2, f(i, k), \ldots, c_j, f(i, k)) \) is generated by \( P \), which is inferred parameters \( v \), and the combined probability distribution of \( X \) is

\[
f(x, P) = \prod_i p_{ij} \quad \text{and} \quad \sum_j p_{ij} = 1
\]

Through the lagrange multiplier techniques, we get
\[
p_y = \frac{E_p \left[ I_{Z(x)\leq y} \sum_i I_{x \in X_i} \right]}{E_p \left[ I_{Z(x)\leq y} \sum_i I_{x \in X_i} \right]} \tag{9}
\]

Therefore, the \( \hat{p}_y \), which is an estimate of \( p_y \), can be written as

\[
\hat{p}_y = \frac{\sum_{k=1}^{N} I_{Z(x)\leq y} I_{x \in X_i}}{\sum_{k=1}^{N} I_{Z(x)\leq y} I_{x \in X_i}} \tag{10}
\]

In this way, we get the probability modifier formula in different states.

### 3.2. The learning strategy of capability unit allocation

The Markov decision model of the manufacturing resource optimization problem by using cross entropy method, it is realized by two steps of iterative process.

1. Adaptive updating of \( \gamma_t \). For a fixed \( \gamma_{t-1} \), let \( \gamma_t \) be a \( \rho \times 100 \% \) percentile of \( Z_{\pi}(X) \) under \( \gamma_{t-1} \). That is \( \gamma_t \) satisfies \( P_{\gamma_{t-1}}(Z_{\pi}(X) \geq \gamma_t) \geq 1 - \rho \) and \( P_{\gamma_{t-1}}(Z_{\pi}(X) \leq \gamma_t) \geq \rho \), where \( X \sim f(\bullet; \gamma_{t-1}) \). A simple estimator \( \hat{\gamma}_t \) of \( \gamma_t \) can be obtained by taking a random sample \( X_1, X_2, \ldots, X_N \) from the pdf \( f(\bullet; \gamma_{t-1}) \), calculating the expected cost of development \( Z_{\pi}(X_k) \) for all \( k \), order them from smallest to biggest as \( Z_{(1)} \leq Z_{(2)} \leq \ldots \leq Z_{(N)} \), and finally evaluating the \( \rho \times 100 \% \) sample percentile as \( \hat{\gamma}_t = Z_{[\lfloor \rho \times N \rfloor]} \).

2. Adaptive updating of \( V_t \). For a fixed \( \gamma_t \) and \( \gamma_{t-1} \), derive \( \hat{V}_t \) from program (10). We use a smoothed updating procedure in which \( \hat{V}_t = \alpha V_t + (1 - \alpha) \hat{V}_t \), \( \alpha \in (0, 0.4, 0.9) \).

### 4. Simulation results

In order to evaluate the effectiveness of the optimal resource allocation strategy, we do simulate and compare the strategy based CE method with the random allocation.

Suppose that a product consists of 8 components. For each component, one of the 5 enterprises will be selected to produce. The level and the cost of each enterprise to complete the task can be obtained through evaluation and analysis of historical production data.

This is abstracted that complex product development task \( F \) can be decomposed to \( F = \{ f_i \mid i = 1, 2, \cdots, 8 \} \). So there are 8 development capability pools and each pool is consisting of 5 development capability units. Evaluation results based on the cost and level of enterprise, we get the development capability unit level set is

\[
LOV = \{lov(c_{ij}) \mid i = 8, j = 5 \} = \begin{pmatrix}
8 & 9 & 8 & 9 & 7 & 7 & 8 \\
7 & 7 & 6 & 7 & 8 & 7 & 7 \\
6 & 8 & 7 & 8 & 9 & 6 & 9 \\
7 & 8 & 5 & 7 & 8 & 6 & 6 \\
8 & 6 & 8 & 8 & 6 & 7 & 6
\end{pmatrix} \tag{11}
\]

And the cost set is
The relationship between probability \( \theta \) and \( \ell_{ij}(c_j) \) is
\[
\theta_{ij} = \frac{\ell_{ij}(c_j)}{10},
\]
that means
\[
\begin{pmatrix}
0.8 & 0.9 & 0.7 & 0.8 & 0.9 & 0.7 & 0.7 & 0.8 \\
0.7 & 0.7 & 0.9 & 0.7 & 0.7 & 0.8 & 0.7 & 0.7 \\
0.6 & 0.8 & 0.8 & 0.7 & 0.8 & 0.9 & 0.6 & 0.9 \\
0.7 & 0.8 & 0.5 & 0.7 & 0.8 & 0.6 & 0.6 & 0.8 \\
0.8 & 0.6 & 0.8 & 0.8 & 0.6 & 0.6 & 0.7 & 0.6
\end{pmatrix}
\]
\( (13) \)

Using \( N = 5000, \rho = 0.16, \alpha = 0.5, d = 10 \), we get the optimal profile is
\[
P_{ij} = \begin{pmatrix}
0.052158 & 0.041300 & 0.032555 & 0.015696 & 0.045621 & 0.040473 & 0.969045 & 0.016020 \\
0.000025 & 0.927973 & 0.043557 & 0.006154 & 0.061264 & 0.000000 & 0.000051 & 0.143709 \\
0.347612 & 0.019945 & 0.923887 & 0.031725 & 0.893115 & 0.000000 & 0.000006 & 0.039158 \\
0.600205 & 0.010782 & 0.000000 & 0.946425 & 0.000000 & 0.944840 & 0.000000 & 0.008904 \\
0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.014679 & 0.143709 & 0.039158
\end{pmatrix}
\]
\( (14) \)

Which means the optimal resource allocation strategy is \( c_{14}, c_{22}, c_{33}, c_{44}, c_{55}, c_{66}, c_{71}, c_{85} \). The optimal imply that, the optimal allocation should be taken is the one that minimizes \( \sum_{i,j} C_{ij}(c_j) / \theta_{ij} \).

In the compare experiment, we do simulate 50 runs for the optimal strategy and random strategy respectively. In random simulation the development capability units are all from the development capability pools and
\[
P_{ij} = \begin{pmatrix}
1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 \\
1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 \\
1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 \\
1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8
\end{pmatrix}
\]

is used, which means randomly select a unit from the operational pool. The comparison results are showing in figure 2 and figure 3.

The abscissa represents simulation time both in figure 2 and figure 3. The ordinate of figure 2 is the total development cost for each process. The ordinate of figure 3 is the number of capability unit allocation times. In figure 2, the average development cost of the optimization strategy in 50 runs of simulation is 3167.1, whereas the average development cost of the random strategy in 50 runs of simulation is 3645.74. In figure 3, the average number of capability unit allocation times of the optimization strategy is 11.38, and the average number of capability unit allocation times of the random strategy is 11.78.

The simulation results show that the optimal strategy could not guarantee the result is better than random strategy for each process. For example, in the 2nd, 8th, 18th and 42nd simulation experiment we find that the total development cost and the number of capability unit allocation times of the optimization strategy are not better than the random strategy. The reason for the phenomenon mentioned above is randomness of development capability unit to complete the task. This randomness has a great influence on the results. Therefore, only by comparing the average could reduce the effect of the randomness. From this view, we can see that the optimal strategy is better than random strategy in the average of development cost and development number.
5. Conclusions
In this paper, considering uncertainty of the product development process, the manufacturing resources optimization allocation problem is studied under the cloud manufacturing environment. The markov decision model of the manufacturing resource optimization problem under the federal collaborative development environment is established. We take the minimum expected cost as the goal, apply the cross entropy method to solve the problem and get an optimal allocation strategy of development capability unit. The simulation results show that through this learning method for the optimal resource allocation strategy is superior to random strategy and capable to improve the efficiency of production.
Acknowledgment
This research is supported by the National Natural Science Foundation of China (Grant No. 51505437), and the Major project of the National Defense Basic Research Program (No. A0920131001).

References
[1] Li B, Zhang L, Wang S, et al. 2010 Cloud manufacturing: a new service-oriented networked manufacturing model Computer Integrated Manufacturing Systems 01:1-7+16 (in Chinese)
[2] Tao F, Zhang L, Guo H, et al. 2011 Typical characteristics of cloud manufacturing and several key issues of cloud service composition Computer Integrated Manufacturing Systems 03:477-486 (in Chinese)
[3] Xu X 2012 From cloud computing to cloud manufacturing Robotics and Computer-Integrated Manufacturing 28(1):75-86.
[4] Tai L, Hu R, Zhao H, et al. 2013 Multi-objective dynamic scheduling of manufacturing resource to cloud manufacturing services China Mechanical Engineering 24(12): 1616-1622 (in Chinese)
[5] Yu J, Li Y, Yu H, et al. 2008 Resources optimization deployment in collaborative manufacturing project based on adaptive ant colony algorithm Computer Integrated Manufacturing Systems 14(3): 576-580 (in Chinese)
[6] Wang S, Song W, Kang L, et al. 2012 Manufacturing resource allocation based on cloud manufacturing Computer Integrated Manufacturing Systems 18(7): 1396-1405 (in Chinese)
[7] Yin C, Zhang Y & Zhong T 2012 Optimization model of cloud manufacturing services resource combination for new product development Computer Integrated Manufacturing Systems 18(7): 1368-1378 (in Chinese)
[8] Yu X, Zhan D, Nie L, et al. 2010 Flexible resource constrained resource leveling project scheduling problem Computer Integrated Manufacturing Systems 09(9): 1967-1976
[9] Laili Y, Tao F, Zhang L, Cheng Y, Luo Y & Bhaba R Sarker 2013 A Ranking Chaos Algorithm for dual scheduling of cloud service and computing resource in private cloud Computers in Industry 64 448-463
[10] Rubinstein RY 1999 The simulated entropy method for combinatorial and continuous optimization Methodology and Computing in Applied Probability 2: 127–190
[11] Boer D, Kroese DP, Mannor S, et al. 2005 A tutorial on the Cross-Entropy method Annals of Operations Research 134(1): 19267
[12] Rubinstein RY 2001 Combinatorial optimization, cross-entropy, ants and rare events. Algorithms and Applications 304–358
[13] Ridder A 2005 Importance sampling simulations of Markovian reliability systems using cross-entropy Annals of Operations Research 134(1):119–136
[14] Hui KP, Bean N, Kraetzl M, Kroese DP 2005 The cross-entropy method for network reliability estimation Annals of Operations Research 134(1):101–118