The Optimization of Resource Allocation Based on Process Mining

Weidong Zhao[1,2(✉)], Liu Yang[1,2], Haitao Liu[1,2], and Ran Wu[1,2]

[1] Software School, Fudan University, Shanghai China
[2] Shanghai Key Laboratory of Data Science, Fudan University, Shanghai China
{wdzhao,12212010021,13212010011,13212010022}@fudan.edu.cn

Abstract. The effectiveness of resource allocation directly affects process performance. In order to optimize resource allocation, this paper proposes a resource allocation model in view of the relationship between resource allocation and process performance, which minimizes process execution time in terms of resource preference, cost constraints and resource availability criteria. Resource coordination is paid less attention in previous resource allocation studies. Therefore, this paper presents the corresponding resource allocation method in consideration of resource coordination, the interval between adjacent activities and distinguishing turnaround time between different resources from event logs. The experiments show that the proposed method can effectively optimize the resource allocation.

Keywords: Resource allocation · Process performance · Process mining · Optimization models · Resource coordination

1 Introduction

Resource allocation, as the key issue of process management, focuses on how to allocate resources efficiently to optimize process performance. Hence, its quality directly affects process performance. Process mining, which can discover useful patterns of resource allocation from process event logs, can optimize the process by eliminating existing bottlenecks in time and is significant in improving both resource execution efficiency and process performance. We only focus on staff (participants) here in view of their special importance [1].

Resource allocation models describe the relationship between process activities and resources. Analysis of resource models can optimize resource allocation. Kumar and Aalst proposed two basic resource allocation modes: push mode and pull mode [2]. The former pushes process tasks to the resources that meet the requirements, while the latter allows resources to request task initiatively from the task pool. The two modes are too simple to deal with complex situations like resource shortage or overload. The purpose of resource allocation models is to allocate the most appropriate resources for process activities [3, 4]. Event logs record the real execution of resource allocation, so mining resource allocation models from event logs is more consistent with actual situations of the process, and more suitable for the optimization of resource allocation. Using the decision

DOI: 10.1007/978-3-319-22053-6_38

tree algorithm, Ly et al. discovered the task allocation rules from historical data and organizational structure [5]. They treated process actors and the type of activities as input, and whether participants are involved in activities as classification results, to learn the resource allocation model inductively. Task allocation rules reflect information on resources such as their preferences, skills, etc. These rules can be used to adjust resource allocation. Huang et al. used association analysis to mine the dependencies among resources [6]. Utilizing the sequence correlation constraint of process activities, they improved the Apriori algorithm and produced two types of resource allocation rules: the resource dependency rule and activity allocation rule. The resource dependency reflected the relationship between resources, which connected the process activities orderly. For example, the resource r_1 performs the activity a_1, then the follow-up activity a_2 will be performed by r_2; activity allocation rules show that some resources frequently participate in a specific task. Resource dependencies reflect the deep interaction between resources, which helps understand the mode of resource cooperation. On the basis of these models, some researchers tried to use certain resource allocation models to achieve automatic resource allocation, thus improving process efficiency. In addition, by mining event logs to analyze resource allocation rules, we also got preliminary exploration in recommending resources to managers [7]. For example, Yang et al. used the hidden Markov model to build resource allocation models, by mining initialization parameters from event logs, to recommend suitable process resources to activities according to the probability of employees involved in these activities and the transaction between staffs [8].

The researches above seldom concerned about the influence that resource allocation has on process performance. Some studies on performance optimization based on resource allocation have been done. Process execution time and cost are two commonly used metrics for process performance. But it is hard to achieve the metrics best simultaneously. As a result, lowering process cost can increase execution time consumption and speeding up process execution might increase cost. To compromise process cost and time, appropriate balance has to be achieved. Kumar et al. proposed a dynamic resource allocation method that balanced process performance and resource access control [9]. Xu et al. presented a resource allocation method, which minimized process cost under limited time constraint [10].

When allocating resources to activities, most methods only consider the applicability, consistency and availability of resources, etc. [11]. However, in real business processes, employees' productivity usually changes due to variations of some external factors such as work stress, business environments and so on. Resources are the key elements of process performance. Process activities and their logical relations are just external form of resource roles and their coordination [12]. Some scholars highlighted the coordination among resources, ensuring the effective coordination among them. For example, Aalst et al. calculated parameters relevant to resource coordination level according to causality between activities [13]. Huang et al. calculated prior probabilities of activities between resources and made judgments of the correlation between resources on the basis of Aalst's researches [14]. Those studies did not pay enough attention to the effect of resource allocation on process performance. What's more, most of them determined resource correlation subjectively or only considered connections between activities, without analyzing the correlation between staff from process event logs. But rich information on resources is hidden in process event logs, which deserves further researches [15].

In order to solve low process performance and the relationship between resources in existing resource allocation models, this paper proposes a new resource allocation model that can minimize the execution time while meeting the requirements of cost and resource availability constraints for higher process performance. On this basis, it fully describes a real execution situation of the process, considering the effect of correlation between resources on resource allocation.

The remainder of the paper is organized as follows: Sect. 2 generally introduces the goal and constraints of the proposed resource allocation model. Sections 3 and 4 describe the way to judge the resource availability, which is the key constraint in resource allocation. Process cost is discussed in Sect. 5. Section 6 touches on resource coordination, and elaborates the resource allocation method aiming at minimal flow time. Experiments are discussed in Sect. 7, and Sect. 8 concludes this paper.

2 Resource Allocation Model to Minimize Total Process Time

The resource allocation table is used to store the status after a resource is allocated. A record is inserted into the table when certain resource is assigned for an activity. Information contained in this record includes the activity v, the assigned resource s, the corresponding role r, the start time of the activity *start time*, the end time of the activity *end time*. The resource allocation table includes the following subjects:

(1) For a business process P, we define the activity set SA = { | i = 1,2, …, k, k is the total number of activities}, resource set SP = { | i = 1,2, …, n, n is the total number of resources} and role set SR = { | i = 1,2, …, m, m is the total number of roles}.

(2) The precursor and sub-sequence relationships between activities and resources: in sequentially connected activities, precursor activity occurs, and then the successor activity occurs after the precursor activity directly. Resources that execute precursor activities are called precursor resources, and while resources that execute subsequence activities are called sub-sequence ones. For example, i is the precursor activity of j, which means j is the subsequence activity of i; If s_i is the precursor resource of s_j, then s_j is the subsequence resource of s_i. Specifically, the process of initiation (termination) activity (resource) does not have precursor (subsequence) activity (resource); there is no precursor or subsequence relationship for parallel activities. In P, RA represents the relationship set among activities, and RP represents the relationship set among resources.

$$\mathrm{P} = (\mathrm{SA}, \mathrm{SP}, \mathrm{SR}, \mathrm{RA}, \mathrm{RP})$$

The aim of resource allocation model is to minimize the total execution time under three constrains: resource preference, resource availability and total cost.

(1) **Resource Preference.** With regard to a particular resource, resource preference is a set of activities that have been executed more often and have higher execution efficiency. Although resources have the ability to perform various tasks, they do

prefer to do some of them. A resource's willingness and efficiency will be much improved if some activities are allocated to them.

(2) **Resource availability.** The availability of resources for activities is limited by simultaneity and workload. Simultaneity indicates that there is no free time to receive new tasks when a resource is executing certain activity.

(3) **Process Total Cost.** Process cost must be limited within a certain range. Time and cost are two of the most important metrics to measure process performance, yet they cannot be optimized at the same time. Minimizing process time blindly is likely to cause the increase of cost, so we need to develop a method that can control cost within a certain range while minimizing process time.

The resource allocation model is defined as follows:

$$D(P) = min \sum\nolimits_{v} Time\,(v) \tag{1}$$

$$\begin{aligned} &\text{preference}(s, v) > \beta, t \in SA, \quad s \in SP \\ &availability(tm, s) = true, \quad s \in SP \\ &\sum\nolimits_{v} cost(s, v) \le \cos t_{\text{lim}}, \quad t \in SA,\ s \in SP \end{aligned} \tag{2}$$

In Eq. (1), *D*(P) is the object function which means to minimize the total time of P, and *Time*(*v*) denotes time spent by each activity *v*. Equation (2) includes three constrains. *preference*(*s,v*) denotes the preference value of *s* to *v,* and *β* is the threshold for eliminating activities that resources are not interested in; *availability*(*tm,s*) denotes whether a resource *s* is available at *tm*; *cost(s,*v) denotes the cost generated when *s is* executing *v,* and $cost_{max}$ is the cost constraint.

3 Resource Preference Constraints

This section shows the measurement for resource preference constraints. Resource preference *preference(s,t)* indicates the priority of *v* for *s*. Higher preference value means that resources are more likely to execute this activity efficiently.

Resource preference support represents the probability that *v* may be executed by *s* in a period, as defined in Eq. (3).

$$\text{support}(s, v) = \frac{\text{count}\left(s, v, tim_1\right) - \text{count}(s, v, tim_2)}{\text{count}\left(v, tim_1\right) - \text{count}(v, tim_2)} \tag{3}$$

where *count*(*s,v,*tim_1) denotes times that *s* has executed *v* until the time tim_1, *count*(*s,v,*tim_2) denotes times that s has executed *v* until tim_2 which is less than tim_1. *count*(*v,*tim_1) count $\left(v, tim_1\right)$ denotes times that *v* has been executed up to tim_1, *count*(*v,*tim_2) represents times that *v* has been executed until tim_2.

Resource preference confidence represents the probability that s will execute t in a period of time, as defined in Eq. (4).

$$\text{confidence}\,(s, v) = \frac{\text{count}\,\left(s, t, tim_1\right) - \text{count}(s, t, tim_2)}{\text{count}\,\left(s, tim_1\right) - \text{count}(s, tim_2)} \tag{4}$$

count(s,tim_1) denotes times of activities that s has executed until tim_1, while *count*(s,tim_2) represents times of activities that s has executed until tim_2.

Resource Preference is related to preference confidence and preference support, as defined in Eq. (5).

$$Preference(s, v) = \alpha\, support(s, v) + (1 - \alpha)\; confidence(s, v) \tag{5}$$

In Eq. (5), α is a parameter ($0 < \alpha < 1$), which represents the weight of resource preference support on resource preference. In order to meet the conditions, *preference*(s,v) has to be greater than the threshold β.

4 Resource Availability Constraints

This section discusses the algorithm of the resource availability constraint function *availability(tm,s).*

(1) ***Resource load***
Resource load represents work pressure of resources. The factors that cause resource overload include the number of activities that has been executed in a certain period (we choose the time window as 1 week) and work time. The *Yerkes-Dodson Law* shows that work efficiency is relevant to work load. With reasonable pressure, resources can work more efficiently, and overload may decrease work efficiency. The curve of the Yerkes-Dodson Law is an inverted U-shape. Inspired by the Yerkes-Dodson Law, resource load can be measured by work efficiency. Since time consumption of v is changing with resources, work efficiency can be calculated by the average time of v divided by time consumption by s. Time consumption start when a resource begins work and end until the start of its subsequence activity. Therefore, through mining event logs, time can be used as the horizontal axis and resource efficiency can be used as the vertical axis to form the curve of resource load. The highest point on the inverted U curve stands for the saturation value of resource load; otherwise, it means that resource efficiency can be improved.

(2) ***Resource potential***
Resource potential signifies the potential of improving work load when the load of resource has already saturated. Higher potential for a resource indicates the larger space to increase the resource capacity, which also means that the resource has the ability to execute more activities. We assume the situation that all the resources are at their saturation value when a new activity arrives. Now we have a young, highly

educated man and an old lady with poor education, whom would you give the task to? It is commonly considered that the young man has more potential, so we allocate the activity to him. The larger the potential is, the higher the probability that a resource can improve its work load.

Clustering analysis, which is a classical data mining method, is used herein to determine the resource potential. Four properties are used to determine the resource potential including age, gender, family status and educational status. Each property has different weights. According to these properties, similarity can be calculated between two resources.

We adopt the K-means algorithm to produce resource groups, which turn out to be three clusters: the high potential set (*hCluster*), the middle potential set (*mCluster*) and the low potential set (*lCluster*) respectively.

(3) ***Resource availability***

Resource availability is used to judge whether the resource can execute a new activity. Three steps are needed to estimate resource availability: (1) Examining whether the resource is busy. If the resource is executing the activity, then the resource is unavailable. (2) Check the workload of unoccupied resources. The resource is also unavailable provided that its load has reached the limit value. (3) If all of the unoccupied resources are overloaded, we'll choose resources from the high potential set (*hCluster*).

5 Total Cost Constraints

In this section, we will illustrate cost constraints function *costValue*.

(1) ***Transmission probability***

$p(tra_{v,v_s}, s)$ represents the probability that v is transmitted to its subsequence v'. Transmission probability can be calculated by the equation *A/B*, where *A* is times that *v* passes to *v'* after being executed by *s*, and *B* is times that *v* is executed by *s*.

(2) ***Total cost prediction***

There may be a risk that the total cost will be beyond the constraint if we keep reducing execution time. So we have to predict the total cost of this process after allocating resources. The prediction of total cost contains two parts: the first is the *certain cost*, which is the cost from the activities that have been assigned for resources; the second is *uncertain costs* resulting from activities that have not yet been assigned. We can calculate the certain cost by summing them up, but we have to predict the uncertain one. The prediction of uncertain costs is defined as Eq. (6).

$$forecastc\,(v) = cost\,(s, v) + \sum\nolimits_{v_s} p(tran_{v,v_s}, s) forecastc(v_s) \tag{6}$$

*forecastc(v),*a recursive equation, denotes the total cost prediction for activities which have not yet been allocated resources. *s* is the resource that take a minimum time consumption of *v*. Because *forecastc*(*v*) is the predicted value, so it allows a deviation compared with the actual value. Meanwhile, resource availability constraints test will spend a lot of time, in order to assure efficiency of the cost constraint test, so we do not take the resource availability constraints test for s. v_s is the subsequence activity selected after v. After predicting the uncertain cost, we can get prediction of the total cost using Eq. (7).

$$overallc\ (s, v) = \sum\nolimits_{v_p} cost\left(v_p\right) + forecastc\,(v) \tag{7}$$

*wh*ere overall *c(s,v)* represents prediction of the total cost after v has been allocated to resource*s* [16]; *cost*(v_p)denotes the cost of activities which has been assigned for certain resources; $\sum\nolimits_{v_p} cost\left(v_p\right)$ denotes the overall cost of all activities from the initiation activity to the precursor of v.

(3) ***Resource replacement ratio***

If the prediction value of overall cost exceeds the cost constraint, we need to reallocate resources to reduce the cost. To decide which activity should be reallocated to the resource, we need to consider whether the activity is included in the longest path. The longest path is the one that takes the longest execution time [10]. The longest path decides execution time of the whole process. In order to avoid extension of execution time during adjustment, it is necessary to find the most suitable replacement activity from both the longest path and other paths, and then compare their influence on execution time after replacement. The activity that has less influence will be chosen.

Before getting the most suitable replacement activity, we need to find out the most suitable replacement resource for each activity in the process. During adjustment, replacing resources will cause modifications of execution time and processing cost. Let $\Delta time$ represent the time difference when resources change from s_1 to s_2, which has less cost than s_1. $-\Delta\text{time}/\Delta\text{cost}$ measures the relative change of cost and time due to resource replacement, which is called the resource *replacement ratio*, denoted as *compensation* $\left(v, s_1, s_2\right)$. Lower because $\Delta\text{cost} < 0$, the resource which has the lowest resource replacement ratio is the best replacement resource. Resource replacement ratio means more reduction of cost with less increase of execution time.

6 Resource Allocation Algorithm

After analyzing resource preference constraints, resource availability constraints and total cost constraints, this section will determine the target of the resource allocation model: the total processing time. Most of researchers only focus on process execution time, ignoring the turnaround time during process execution. Turnaround time is the time from the end of the precursor activity to the start of its subsequence activity. In general, the turnaround time of neighboring activities varies mainly due to the different collaboration relationship between resources. Resources with higher collaboration level usually have shorter turnaround time. To simplify the problem, this paper considers that the collaboration between resources is the main reason for turnaround time difference.

For the reason that we have considered the turnaround time caused by resource collaboration level, we achieve the allocation goal D(P). We need ensure that the execution time *operatetime(v)* for each activity v in the process is as little as possible, while the turnaround time with precursor activities *turntime(v)* also needs to be controlled. Now D(P) has become a multi-objective programming problem. One way to solve the multi-objective programming problem is using linear weighted method to transform the multi-objective programming problem into a single objective programming problem.

In run-time, turnaround time between different resources may vary greatly. From events logs, we find that turnaround time always has larger impact on the total time than process execution time. In some instances, the difference between turn-around time and execution time may reach an order of magnitude, considering the sum of turnaround time in a process may be 30 h while execution time may only be 3 h. A greater order of magnitude that turnaround time is compared to execution time will enhance the impact of turnaround time on process performance. Here is a measurement method using magnitude level k.

$$k = \lg \frac{\overline{tTime}}{\overline{oTime}} \tag{8}$$

where k represents difference order of magnitude between turnaround time and execution time in process instances. $\overline{tTime}$ denotes the average value of turnaround time of completed instances, and $\overline{oTime}$ denotes average value of execution time of process instances. Larger value of k indicates that the turnaround time has greater effect on the total time. For example, turnaround time with magnitude level of 3 has a greater impact on the total time than magnitude level of 1. So in order to embody the impacts of turnaround time on process performance caused by different magnitude level, the weight of execution time is set to $\frac{1}{\text{k}+1}$ and the weight of turnaround time is set to $\frac{k}{k+1}$. Meanwhile, in order to achieve the same order of magnitude as that of execution time, we let the turnaround time multiply 10^{-k}. The time spent on each activity v of the resource allocation model's target $D\,(\text{P}) = min \sum_v time\,(v)$ is:

$$time(v) = \frac{1}{k+1} * operateime\,(v) + \frac{k}{k+1} * 10^{-k} * turntime\,(v) \tag{9}$$

In Eq. (9), *operatetime*(v) is time consumption of each activity, and *turntime*(v) is turnaround time between activities and their predecessors.

Collaboration among resources is regarded as mutual adjustments and learning processes between resources. Researches on human learning have shown that the learning curve is a power function curve. The learning curve indicates that more times you study, less time you consume. At first the study rate is relatively fast, but it will gradually diminish to be flat [16]. From this point of view, we can conclude that resources will have relatively low level of correlation and high level of turnaround time at the early stage of cooperation. With the enhancement of correlation, turnaround time decreases and tends towards stability.

On the basis of the learning curve, a regression equation for turnaround time prediction is proposed.

$$turntime = startime * tms^{-r} \tag{10}$$

The *startime* in Eq. (10) is turnaround time at the first collaboration stage between resources; *r* represents collaboration coefficient between resources; *tms* represents times that two resources collaborate; *turntime* is turnaround time provided that collaboration times of these resources are *tms*. Firstly, evaluating the logarithm of both sides, and then nonlinear regression analysis will be converted into linear regression analysis. We can get a linear regression equation in which lg(*turntime*) is the dependent variable and lg(*tms*) is the independent variable. By using the sum of squared error and taking the derivative of *r*, the model will attain the best fitting state that the sum of n deviation is minimum.

$$r = \frac{\lg(starttime)\sum_{i=1}^{n}\lg(tms)_i - \sum_{i=1}^{n} lg(tms)_i * lg(turntime)_i}{\sum_{i=1}^{n}[lg(tms)_i]^2} \tag{11}$$

We can get the collaboration coefficient *r* by inputting the corresponding data of event logs into Eq. (11), and then predict turnaround time of next collaboration. Not only resources' learning abilities are the determinants of collaboration coefficient *r*, but also resources' cooperation, operation capacities and even information systems. The collaboration coefficient *r* will turn out to be larger if the resources have strong cooperation abilities or are familiar with business operations. A good management system and highly efficient staff will lead to larger *r* as well. Hence, the collaboration coefficient *r* represents resources' comprehensive abilities.

The resource allocation algorithm of minimum execution cost was put forward by Xu et al., which was to find all the resources that have the lowest execution cost for activities in the process at first, then detect the availability of these resources, and finally check whether the total time of the process exceeds the constraint value after ensuring availability [10]. This method may make the whole process repeatedly. A new method that can detect the availability and cost constraints while allocating resources to each activity is proposed in this section. The steps to allocate resources for each activity are shown as follows:

(1) Find the resource that has the least execution time when performing the target activity *v*.
(2) Detect whether the resource satisfies the constraints of resource preference. Looking for candidate resources, which have less execution time and satisfy the resource preference constraint.
(3) Detect the availability constraint of the resource given by step (2). If the resource is unavailable, then it will be replaced by other suitable resources, which meet the availability constraint.
(4) Predict the total cost after the target activity has taken this resource, and test whether it exceeds the cost constraint to find the most suitable activity.
(5) If the resource has been replaced in step (4), then repeat step (3) and step (4) until we find out the resource that satisfies both the availability constraint and total cost constraint.

7 Experiments

Some experiments are conducted to show effectiveness of the method proposed above. In the first place, we check whether it is effective to treat resource collaboration as a learning procedure. We have collected some airline compensation process logs and observe variance of average flow time as Table 1.

Table 1. Flow time variance as the number of instances increases

Number of instances	30	40	50	60	70	80	90	100
Average flow time (h)	121	103	90	82	77	75	80	75

It is showed that flow time decreases very fast when the number of process instances increases from 30 to 100, and after that flow time slightly fluctuates in Table 1. The reason is that resource collaboration level improves with increase of cooperation times. Thus, turnover time and flow time will be reduced. However, collaboration capability would remain stable after the number of instances reaches 70. Note that flow time rises a little after the number of instances is 100, which may be explained by the fact that some activities cannot be executed by suitable resources due to the availability issue.

We also check improvement in terms of flow time considering resource collaboration. We choose 8 process log sets of the airline company randomly. The number of process instances of each set is 30. After extracting the original logs, data pre-processing is conducted to filter out resources in which interaction frequencies are less than 3. The remained process instances are used for analysis. A small number of cooperation activities are insufficient to measure resource collaboration. Then, we use the algorithm that only concern $operatetime(v)$ in $time(v)$ without touching on collaboration. In comparison, we use the resource allocation algorithm proposed in this paper taking account of collaboration as allocation guide. To compare the effect of collaboration, the measure $time(v)$ and $operatetime(v)$ are used respectively. The average flow time is used as performance metric. Figure 1 depicts the results.

It can be seen that process execution time reduces significantly by taking resource collaboration into account. In addition, we can see more satisfactory results as the number of process instances increases. The underlying reason is that more event logs can generate more accurate collaboration measurement, which in turn provides more effective allocation guideline. It can be noted that improvements can still be made for process data with fewer event records through deleting some process logs. Even a small number of event records can provide sufficient evidence for resource collaboration.

We use the same process log sets as Fig. 1 and compare the resource allocation algorithm proposed in this paper considering resource coordination capacity with the algorithm (named rb) by Kumar [17]. The clearest difference between these two algorithms is that the former predicts collaboration capacity between resources using regression analysis, and considering collaboration to be dynamic as time goes. But the latter calculated the average value as collaboration capacity. It did not pay attention to

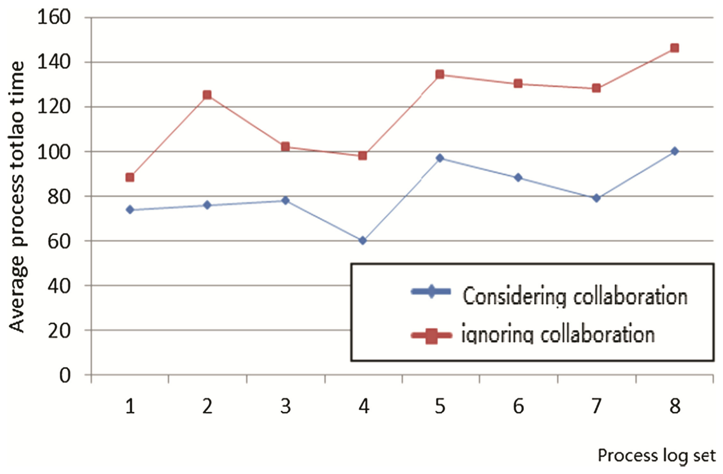

Fig. 1. Considering vs. ignoring collaboration between resources

changing patterns of collaboration between resources, thus lacking of considering collaboration.

Huang also proposed a method (named mc) to compute the degree of collaboration between resources. The main idea of this method was based on the premise of guaranteeing the correlation between resources, and measuring the strength of collaboration between resources by calculating the probabilities [15]. In Fig. 2, process log sets are denoted as the horizontal axis, and the average process total time is denoted as the vertical axis. We can see that dynamic collaboration capacity between resources have more positive effects on performance than the static one.

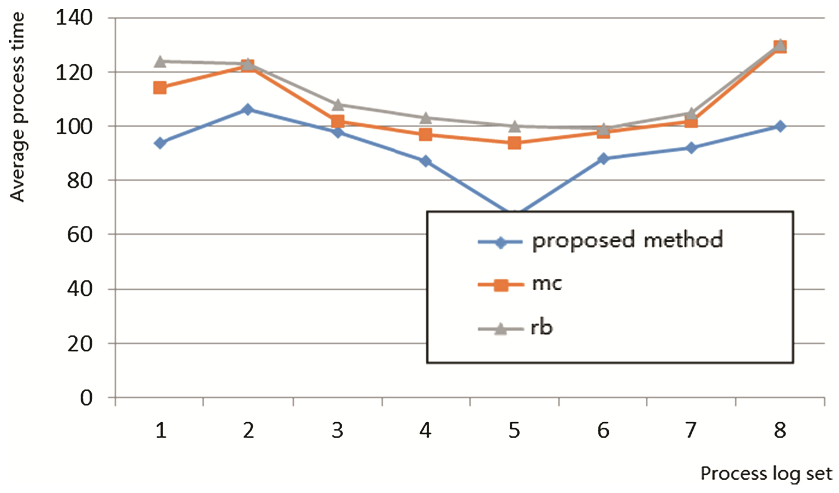

Fig. 2. The influence of different methods in considering collaboration between resources

8 Conclusions

Resource allocation is an important issue in the field of process management. The quality of resource allocation not only determines how the process is executed, but also affects process performance. Process mining has provided new insights and methods for resource allocation. This paper discusses the resource allocation method from the perspective of their effects on process performance, ensuring the process takes less time under the cost constraint. Moreover, this paper also analyzes the effects of collaboration cohesion between resources on process performances. We will do further researches on the coordination among resources, especially on the influence of deep collaboration among resources, in which we will mine further insights from process event logs.

Acknowledgments. Shanghai Pujiang Program no. 14PJC017 and the National Nature and Science Foundation of China under Grants no. 71071038 support this work.

References

1. Van Hee, K., Serebrenik, A., Sidorova, N., et al.: Scheduling-free resource management. Data Knowl. Eng. **61**(1), 59–75 (2007)
2. Wang, J., Kumar, A.: A framework for document-driven workflow systems. In: van der Aalst, W.M., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 285–301. Springer, Heidelberg (2005)
3. Russell, N., van der Aalst, W.M., ter Hofstede, A.H., Edmond, D.: Workflow resource patterns: identification, representation and tool support. In: Pastor, Ó., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 216–232. Springer, Heidelberg (2005)
4. ZurMuehlen, M.: Organizational management in workflow applications–issues and perspectives. Inf. Technol. Manag. **5**(3–4), 271–291 (2004)
5. Ly, L.T., Rinderle, S., Dadam, P., Reichert, M.: Mining staff assignment rules from event-based data. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 177–190. Springer, Heidelberg (2006)
6. Huang, Z., Lu, X., Duan, H.: Mining association rules to support resource allocation in business process management. Expert Syst. Appl. **38**(8), 9483–9490 (2011)
7. Senkul, P., Toroslu, I.H.: An architecture for workflow scheduling under resource allocation constraints. Inf. Syst. **30**(5), 399–422 (2005)
8. Yang, H., Wang, C., Liu, Y., Wang, J.: An optimal approach for workflow staff assignment based on hidden Markov models. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM-WS 2008. LNCS, vol. 5333, pp. 24–26. Springer, Heidelberg (2008)
9. Kumar, A., Van Der Aalst, W.M.P., Verbeek, E.M.W.: Dynamic work distribution in workflow management systems: how to balance quality and performance. J. Manag. Inf. Syst. **18**(3), 157–194 (2002)
10. Xu, J., Liu, C., Zhao, X.: Resource allocation vs. business process improvement: how they impact on each other. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) BPM 2008. LNCS, vol. 5240, pp. 228–243. Springer, Heidelberg (2008)
11. Reijers, H.A., Jansen-Vullers, M.H., ZurMuehlen, M., Appl, W.: Workflow management systems+swarm intelligence=dynamic task assignment for emergency management applications. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 125–140. Springer, Heidelberg (2007)

12. Bozkaya, M., Gabriels, J., Werf, J.: Process diagnostics: a method based on process mining. In: Information, Process, and Knowledge Management, pp. 22–27. IEEE (2009)
13. Van Der Aalst, W.M.P., Reijers, H.A., Song, M.: Discovering social networks from event logs. Comput. Support. Coop. Work (CSCW) **14**(6), 549–593 (2005)
14. Huang, Z., Lu, X., Duan, H.: Resource behavior measure and application in business process management. Expert Syst. Appl. **39**(7), 6458–6468 (2012)
15. Van der Aalst, W.M.P.: Discovery Conformance and Enhancement of Business Processes. Springer, Heidelberg (2011)
16. Bellman, R.E.: Dynamic Programming. Princeton University Press, Princeton (1957)
17. Kumar, A., Dijkman, R., Song, M.: Optimal resource assignment in workflows for maximizing cooperation. Bus. Process Manag. **8094**, 235–250 (2013)