International Journal of Engineering

Journal Homepage: www.ije.ir

Cycle Time Reduction and Runtime Rebalancing by Reallocating Dependent Tasks

M. Yaghoubi*[a,b], M. Zahedi[b]

[a] *Computer Engineering Department, Faculty of Engineering, Golestan University, Iran*
[b] *Computer & IT Engineering Department, Shahrood University of Technology, Shahrood, Iran*

PAPER INFO

Paper history:
Received 24 April 2017
Received in revised form 09 July 2017
Accepted 08 September 2017

Keywords:
Business Process
Cycle Time
Workload Balancing
Resource Allocation
Dependent Tasks
Task Concurrency

ABSTRACT

Business Process Management Systems (BPMS) is a complex information system that provides designing, administrating, and improving the business processes. Task allocation to human resources is one of the most important issues which should be managed more efficiently in BPMS. Task allocation algorithms are defined in order to meet the various policies of organizations. The most important of these policies could be reducing the average cycle time, balancing the resource workload, increasing the product quality and minimizing the production costs. Therefore, choosing an appropriate resource in task allocation algorithms could influence on overall policy of the organization. In heavy load conditions or when the number of human resources is limited, workload balancing can increase the stability of the system. In this paper, a task allocation algorithm is proposed to rebalance the resource workload in runtime while minimizing cycle time by offering dependent pair tasks to resources for concurrent processing. The experimental results show that the combination of previous algorithms with the proposed algorithms would have 4.42% reduction in cycle time in contrast to most efficient state-of-the-art algorithms.

doi: *10.5829/ije.2017.30.12c.03*

1. INTRODUCTION

Development and productivity of many companies and organizations are growing rapidly. Although this development is achieved through the use of novel technology and advanced skills in management, the strong competitions in the global market cannot be avoided. Companies and organizations which are working in a field of global trade should benefit from new ways of customer orientation and management of business processes to retain and increase their customers. Therefore, BPMS provides to better manage of the business processes [1-3]. BPMS makes for users to have an intelligent management on business processes. While the older tools of workflow management systems (WFMS) provide development, execution and management of business processes, BPMS can also provide the possibility to manage the interaction between processes and it is more compatible with process models with reality [3, 4]. In this study, we propose that the dependent tasks to be distributed according to expertise and ability of resources by creating balance in the workload of service agents (resources) and managing diversity and dependency of tasks, which are assigned in worklist of each resource. Appropriate allocation of tasks alongside balancing the workloads can reduce the delivery time (cycle time) of requests in the business process. To achieve this goal, we use the history of process executions by looking events log and provide a more appropriate distribution of tasks to the work list of resources by extracting information of *"concurrent performing of tasks"*. The main idea of this article is based on statistical analysis of the history of events log from previous performances of the business process. While techniques such as modeling on queuing networks [5-9], reinforcement learning models [10] and stochastic models [8, 11, 12] have been used in many studies conducted.

*Corresponding Author's Email: *m.yaghoubi@shahroodut.ac.ir* (M. Yaghoubi)

2. RELATED WORKS

System stability in the business process execution is one of the main factors in BPM systems as a type of queuing network systems, which is actually controlling and maintaining the length of the worklists of resources from not violating a specified size of usually a long given time [13]. A technique to increase the system stability is the resource workload balance, especially when the number of service provider resources is limited. This technique is used in various fields in industry and computer science. Among them, distributed systems and parallel computing [14], grid computing [15] and database management system [16] could be noted. The technique has also been used in workflow management systems [10, 17-27].

Annually, costly researches are done to reduce the cycle time in manufacturing plants; one of them is the research carried out on open issue of Job shop scheduling, which aims to create an optimal scheduling for minimizing flow time [28-30]. The result of this research can be used in the business process management systems and while the problems defined in BPMS become complex due to the probabilistic nature and its interaction with human activities.

In general, a major factor in the efficiency of implementing business processes can be known in the way of task allocation to resources. FIFO (First in First Out), SPT (Shortest Processing Time), EDD (Earliest Due Date) and MST (Minimum Slack Time) are among the simplest techniques provided [31]. The advantage of these basic methods is that they can be easily used in any system or combined with other more complex methods. Another factor which can be raised is the use of ability and expertise information of resources during the assigning tasks to agents [10, 17, 18, 32]. This approach indirectly tries to create a kind of priority in task allocation according to role of resources, which collectively improves the performance of allocation algorithm in the selection of appropriate resource, but it decreases the flexibility of allocation algorithm in free choice of human resources.

2. 1. Cycle time Reduction as an Objective There exist some papers which considered cycle time reduction as their objective [17, 21, 33]. Xingmei Liu et al. [33] showed their technique reduces the cycle time by extracting social relation between resources involved with each trace as a parameter in reinforcement learning algorithm for task allocation problem. They have examined their own algorithms on BPI Challenge 2012 dataset. They showed that considering social relations lead to cycle time reduction [33-35]. Muehlen et al. [21] have outlined the major aspects of resource management within workflow applications, following the workflow life cycle. They presented a generic resource Meta model for the initial specification of the resource structure and its population, which combines workflow-oriented with organization-oriented modeling concepts. Finally, they presented strategies for the use of workflow audit trail data to improve resource utilization and develop new process strategies. Xu et al. [17] presented a resource allocation method, which minimized process cost under the limited time constraint. A resource selection rule is presented in the study of Nisafani et al. [23] which considers factors such as workload, queue, working hours, inter-arrival time, and others which affect human resource performance. They used a Bayesian Network (BN) to incorporate those factors into a single model, which we have called the Bayesian Selection Rule (BSR). They show that the BSR can reduce waiting time, completion time and cycle time. The study of Wibisono et al. [24] proposes an on-the-fly performance-aware resource allocation to manage human resources in BPM. They utilize Naïve Bayes Model in the Naïve Bayes Selection Rule (NBSR) algorithm in order to select an appropriate resource to perform an incoming task. The study of Kumar et al. [34] introduced a novel and more sophisticated mechanism to distribute work than the ones that are currently employed in workflow systems. Compared to existing workflow management systems which provide pure *push* or *pull* approaches, the mechanism allows on-the-fly balancing of quality and performance considerations. It merges work distribution and delegation into an integrated framework.

2. 2. Workload Balancing as an Objective There are some papers which considered workload balancing as their objective [17, 21, 33]. Although cycle time reduction is an important issue in BPM, some workload balancing mechanisms have emerged to resolve the limitation of resources' abilities [25, 35].

The papers presented by Byung-Hyun Ha et al. [22] showed that the balancing of resources' workloads makes it more resistant to the increased volume of customers' incoming requests. Larbi et al. [35] presented a model to solve scheduling problem in identical parallel machines. They solve load balancing in parallel machines by mapping strategy to time Petri-net. Byung-Hyun Ha et al. [25] presented a method for workload balancing of agents that perform the tasks in the business process. They used an analytic process model based on process specification and execution history data. They then used a queuing network built from the analytic process model in order to establish task assignment policies for the efficient execution of the business process. After that, as a practical use of the queuing network, workload balancing is presented to improve overall business process efficiency using a linear programming formula. Finally, a set of simulation experiments is conducted in order to validate the performance with respect to using shared work lists [22, 25].

2. 3. Business Process Models Resource allocation problem in BPMS is a chain of decision problems which is modeled as Markov Decision Processes (MDPs) and queuing network in the literature [10, 25, 33]. Since resource allocation in BPMS is an interactive problem, reinforcement learning would be an appropriate option to solve the corresponding MDP problem. Ly et al. discovered the task allocation rules from event log and organizational structure by using the decision tree algorithm. They considered process performers and type of activities as input, and whether participants are involved in activities as classification results, to learn the resource allocation model inductively. They concentrated on a process mining from resource perspective such as staff assignment rules extraction. They have shown the problem of deriving staff assignment rules using information from audit trail data and organizational information. Ly et al. [36] have used decision tree learning to derive meaningful staff assignment rules. Thus, it is possible to provide staff assignment information about activities enabling a better understanding of the underlying process. Yang et al. [37] 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 among the staff.

In many studies conducted, stochastic models are used as a model to analyze business processes. This research has shown that these stochastic models can be used for various purposes very well. Probabilistic Timed Graph which was established on the basis of plausible events in business processes, showed that it can have a better analysis from the performance of resources and thus has provided better management for scheduling the resources [38]. Using the queue modeling in BPMS and taking advantage of simulation techniques, Narahari et al. [8] could achieve the various methods to reduce cycle time by using queuing model analysis. Using the modeling of each task in the business process as a queue service provider, Son and Kim [9] could provide a method to calculate the capacity of each resource due to completion time for the process execution. Chang et al. [5] studied methods to identify the critical path in business processes. Stochastic Workflow Net (SWN) is one of the other models which are widely used by researchers. This model has been well-defined based on theoretical foundation. SWN is a stochastic counterpart of untimed Petri Nets which is used for modeling the workflow and its analysis results are invoked to business processes [8, 11, 12, 39].

3. THE PROPOSED METHOD

Some concepts and definitions such as analytic process model, stored event, task, work item and activity in BPMS are introduced in order to clarify the proposed method.

3. 1. Analytic Process Model In this section, we suggest Analytic Process Model, which serves as an intermediate model from the business process model to process in the queueing model. The analytic process model provides the information about the business process flow and the resources executing tasks, so we define the analytic process model as an n-tuple $p = \langle \varphi, T, F, R, U, \mu, \delta, Q, wl \rangle$ which is characterized by the following:

i. $\boldsymbol{\varphi}$ is arrival rate of customers' incoming requests.
ii. $\boldsymbol{T}$ is a set of defined tasks in the business process.
iii. $\boldsymbol{F} \subseteq \boldsymbol{T} \times \boldsymbol{T}$ is a set of directional connections between tasks, which represents the defined workflow in the business process and determines what task should be done after other task.
iv. $\boldsymbol{R}$ is a set of human resources.
v. $\boldsymbol{U} \subseteq \boldsymbol{T} \times \boldsymbol{R}$ is a set of responsibility of resources to carry out the tasks. Each member of the set indicates which resource is responsible for which task in the business process.
vi. $\boldsymbol{\mu}: \boldsymbol{U} \to \mathbb{R}^{+}$is a function, which maps each item $\langle t, r \rangle \in U$ onto execution time of task t that are performed by resource r.
vii. $\boldsymbol{\delta}: \boldsymbol{U} \times \boldsymbol{T} \to \mathbb{R}^{+}$is a function which maps each item $\langle t, r, t' \rangle \in U$ onto concurrent execution time of two dependent tasks of t and t' that are perfoemd by resource r.
viii. $\boldsymbol{Q}: \boldsymbol{R} \to \boldsymbol{T}^{*}$ is a function which associates resources to a queue of work items in terms of sequences.
ix. $\boldsymbol{wl}: \boldsymbol{R} \to \mathbb{R}^{+}$ is the amount of workload of each resource. The value of $wl(r)$ requires $\boldsymbol{Q}$ and $\boldsymbol{\mu}$ components and is computed by $wl(r) = \sum_{t \in Q(r)} \mu_{t,r}$.

3. 2. Stored Event A 5-tuple of $e = \langle Type, CaseId, Task, Time, ResourceId \rangle$ is an event stored in the event log which is recorded in system storage during the *creation*, *allocation*, *start* and *finish* of each work item.

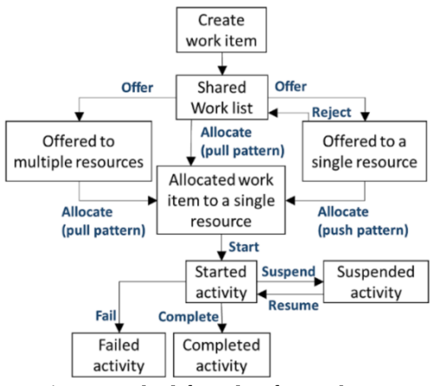

Figure 1. The lifecycle of a work item

i. *Type:* Specifies the type of event which takes one of the values: *allocation*, *Start*, *Complete, Fail, Cancel, Suspend* and *Resume* by itself.
ii. *CaseId*: is the case identifier that event *e* occurs on it.
iii. *Task:* is the work item on which event has been occurred while performing.
iv. *Time*: is timestamp of event.
v. *ResourceId*: is the resource identifier which has performed the task related to event *e*.

Figure 1 shows life cycle of a work item in a state diagram. Each item in the figure represents a state of a work item which may occur in life cycle. Each edge shows an occurrence of an event in which the state of a work item changes and is then recorded in the log.

In this paper, we show an event with symbol *e* and use the functions *e.type* and *e.case* respectively to access the *Type* and *Case* that the event occurred on it. Similarly, we can use the operator dot(.) to access the other components of event.

3. 3. Task, Work Item and Activity The three concepts of task, work item and activity are the main concepts in BPMS. Figure 2 depict the conceptual relationship between the mentioned concepts. When a task in an instance of process execution is attributed to a case, a work item is created; in other words, a work item is a task which defines on a certain case; when a work item is assigned to a certain resource to perform the task on the case, an activity is defined. Execution of activity *a* in BPMS, a sequence of events is recorded in Event log. Each event *e* includes information of case, resource and task which is accessible with three functions *e.case*, *e.resourceId* and *e.task*, respectively. These values are the same for all events of an activity, then we use three functions *a.case*, *a.resourceId* and *a.task* for each activity to easier access to this information. There is at least one event with type '*Start'* and one event with type *'Complete'* in each completed activity which initialize the values of *a.start* and *a.complete*. So, we have Equation (1) to initializing them by the following:

$$\begin{aligned}&\exists e \in Event\ log, e.Type = Start, \boldsymbol{e}\ occurs\ on\ performing\ \boldsymbol{a} \Rightarrow a.Start = e.Time\\ &\exists e \in Event\ log, e.Type = Complete, \boldsymbol{e}\ occurs\ on\ performing\ \boldsymbol{a} \Rightarrow a.Complete = e.Time\end{aligned} \tag{1}$$

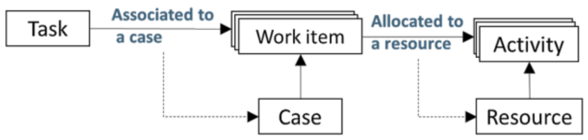

Figure 2. Conceptual relationships between task, work item and activity

3. 4. Estimating the Execution Time of Tasks By analyzing recorded events in the Event-log, there is the possibility to calculate average execution time of each task in business process. This value is good to estimate for execution time of the activity. Equation (2) shows calculation method for execution time of performing activity *t* by resource *r*.

$$\mu^{r}(t) = \frac{\sum_{a \in AC^{t,r}}(a.Complete - a.Start)}{|AC^{t,r}|} \tag{2}$$

$AC^{t,r} = \{a | a \in EventLog, a.Task = t, a.resourceId = r\}$

The event log of a real-life process of BPI Challenge 2012, adopted from a Dutch financial institute, is used for simulation. Of 6078 cases, 5 activities and 55 resources are acquired from the real-life log. By investigating BPI challenge 2012 dataset, it is observed that 4823 pair activities are simultaneously executed by one resource. By more analyzing the pair activities, this comes out that they are relevant from data perspective and the resources probably intentionally choose to process the pair activities so as to process faster. The fact is also revealed that some pair activities take less time when processed simultaneously than individually. So what is obvious is the impact of conducting these activities on each other that will be discussed later in this article. Here, we introduce two types of estimating the execution time for tasks.

3. 4. 1. Estimating the Execution Time of Independent Tasks In Equation (2), if there has not been performed any other activity by the same resource concurrently with all activities $a \in EventLog$ conducted by resource *r*, the value calculated in Equation (2) will be the value of *"estimation of execution time for independent tasks"*.

3. 4. 2. Estimating the Execution Time of Dependent Tasks Consider the two tasks of t_1 and t_2, and set of all activity in event log related to these two tasks which are performed by resource *r*, has been shown with $A^{r}_{t_1} = \{a \in EventLog | a.Task = t_1, a.ResourceId = r\}$ and $A^{r}_{t_2} = \{a \in EventLog | a.Task = t_2, a.ResourceId = r\}$ respectively. For each pair of activities $a_i \in A_{t_1}$ and $b_j \in A_{t_2}$, two activities a_i and b_j may be done independently or concurrently for different values of *i* and *j* by resource *r*. Simultaneity of two activities may occur in one of the following ways.

i. Start and completion time of an activity a_i is absolutely between start and completion of b_j.
ii. Part of the first or last of an activity to be inside the other; in other words, starting time of a_i is within the start and completion time of b_j, or starting time of b_i is within the start and completion time of a_j

So, set $A_{t_1}\varphi A_{t_2}$ is a subset of $A_{t_1} \times A_{t_2}$ which includes all concurrent activities of two tasks t_1 and t_2 which is defined based on Equation (3).

$$A_{t_1}^r \varphi A_{t_2}^r = \left\{\begin{matrix} \langle a,b\rangle | a \in A_{t_1}^r, b \in A_{t_2}^r, \\ a.Start \le b.Start \le b.Complete \le a.Complete \text{ or} \\ b.Start \le a.Start \le a.Complete \le b.Complete \text{ or} \\ a.Start \le b.Start \le a.Complete \le b.Complete \text{ or} \\ b.Start \le a.Start \le b.Complete \le a.Complete \end{matrix}\right\} \quad (3)$$

Statement $A_t^r \varphi A_{t_k}^r = \emptyset, \forall t_k \in T$ is established for each independent task *t* in event log. For a dependent task t', we should obtain its independent ($inA_{t'}^r$) and dependent ($deA_{t'}^r$) activities. Then, we estimate the dependent and independent execution time. Since $A_{t'}^r = inA_{t'}^r \cup deA_{t'}^r$ is established, obtaining one of them is sufficient. Equations (4) and (5) have shown how to calculate them.

$$deA_{t'}^r = \{a | a \in A_{t'}, \exists t \in T, \exists b \in A_t, \langle a,b\rangle \in A_{t'}\varphi A_t\} \quad (4)$$

$$inA_{t'}^r = A_{t'}^r - deA_{t'}^r \quad (5)$$

To estimate the independent execution time of a dependent task t', it is sufficient that we use $a \in A_{t'}^{in}$ instead of the $a \in EventLog$ in Equation (2). As a result, independent time estimate of a dependent task t' is obtained from Equation (6).

$$\overset{r}{\mu}(t') = \frac{\sum_{a\in AC^{t',r}}(a.Complete - a.Start)}{|AC^{t',r}|}$$
$$AC^{t',r} = \{a | a \in inA_{t'}^r, a.Task = t, a.resourceId = r\} \quad (6)$$

For a dependent task t', there are also execution times dependent on another task *t*. To estimate, it is necessary that functions of $T_+, T_\cup$ and $T_\cap$ to be defined for two activities *a* and *b* so that we can estimate the dependent execution time t' based on Equation (10).

$$T_+(a,b) = (a.Complete - a.Start) + (b.Complete - b.Start) \quad (7)$$

$$T_\cap(a,b) = \min(a.Complete, b.Complete) - \max(a.Start, b.Start) \quad (8)$$

$$T_\cup(a,b) = T_+(a,b) - T_\cap(a,b) \quad (9)$$

$$\delta^r(t',t) = \frac{\sum_{\substack{\langle \boldsymbol{a},\boldsymbol{b}\rangle \in A_{t_1}^r \varphi A_{t_2}^r \\ a.resourceId = b.sourceId = r}} (T_\cup(a,b))}{Number\ of\ element\ in\ \Sigma} \quad (10)$$

In Equation (10), function $\delta^r(t',t)$ is *'estimated dependent execution time'* of dependent task t' when it is conducted concurrent with task *t* by resource *r*. If the value of $\delta^r(t',t)$ is smaller than the value of $\overset{r}{\mu}(t') + \overset{r}{\mu}(t)$ for a dependent task t', it shows that if resource *r* performs task *t* concurrent with task t', it has saved in time of doing tasks *t* and t' in amount of $\overset{r}{\mu}(t') + \overset{r}{\mu}(t) - \delta^r(t',t)$. On the other hand, if $\delta^r(t',t)$ is larger than the value of $\overset{r}{\mu}(t') + \overset{r}{\mu}(t)$, it is better that dependent task t' is performed independently by resource *r* to improve the cycle time. Another important point is that a task may be independent for a resource and to be dependent for another resource and values $\overset{r}{\mu}(t') + \overset{r}{\mu}(t)$ and $\delta^r(t',t)$ to be different according to the resource.

3. 5. Workload Rebalancing by Reallocating the Dependent Tasks

Workload balancing of resources and cycle time reduction of business process are the most important goals of BPMS. If the objective of workload balancing is sought in a system, the second objective which is the optimum value for the cycle time, will not be achieved very well. And on the other hand, creating an optimum cycle time will lead to non-balancing of workload [32]. When the policy of workload balancing is followed in a system, the situation of cycle time may be suboptimal for several reasons. These reasons include:

i. Reducing the cycle time has not been seen theoretically in equations of workload balancing
ii. Workload balancing is performed by estimated values from service time of resources and tasks are placed in specific work list of resources based on estimated values. But in fact, a work list may be finished earlier and resources become inactive for some time.

Byung-Hyun Ha et al. [25] could significantly improve the cycle time of cases with the workload estimation of resources based on queuing network model and providing new workload balancing algorithm.

We follow the same objective and we want to compensate the weakness of workload balancing algorithm by reducing cycle time. In this paper, we develop the algorithm presented in [25] and provide heuristic distribution of work items which reduces the cycle time of cases by offering synchronization of dependent tasks, in addition to balancing the workload. The proposed algorithm is shown as follow:

Algorithm: Workload rebalancing by reallocating dependent tasks (WRRDT)
Input: business process model $p = \langle \varphi, T, F, R, U, \mu, \delta, Q, wl\rangle$

```
1: $\overline{wl} \leftarrow \sum_{r\in R} wl(r) / |R|$

2: $LWR \leftarrow \{r | r \in R, wl(r) < \overline{wl}\}$ // Resources with workload lower than average workload
4: for each ($r \in LWR$){
5:     $T_r \leftarrow \{t | \langle t,r\rangle \in U\}$
6:     $max \leftarrow -\infty$
7:     $A_{selected} \leftarrow \emptyset$
8:     if $\left(\begin{matrix}\exists t,t' \in T, \exists\langle a,b\rangle \in A_t^r \varphi A_{t'}^r, \\ \mu^r(t') + \mu^r(t) - \delta^r(t',t) > 0\end{matrix}\right)${
9:         for each($\langle a,b\rangle \in A_t^r \varphi A_{t'}^r$){
10:            if ($\mu^r(t') + \mu^r(t) - \delta^r(t',t) > max$){
11:                $a_{selected} \leftarrow a$
12:                 $b_{selected} \leftarrow b$
13:                 $max \leftarrow \mu^r(t') + \mu^r(t) - \delta^r(t',t)$
14:             }//if
15:         }//for each
16:         $A_{selected} \leftarrow \{a_{selected}, b_{selected}\}$
17:         $Q^r \leftarrow A_{selected} \cup Q^r$ //reallocate to resource r
18:     }//if
19:     else if ($\exists t \in T_r, \exists a \in A_t^r$ and a not startted )
20:         $Q^r \leftarrow Q^r \cup \{a\}$
21: }//for each
22: $\overline{wl} = \sum_{r \in R} wl(r) / |R|$

23: $LWR \leftarrow \{r | r \in R, wl(r) < \overline{wl}\}$
24: }//while
```

Suppose $|R|$ is the number of resources and $|T|$ is the number of tasks defined in the process and C is the number cases in a given period of time in the event log, they are needed to compute $A_t^r \varphi A_{t'}^r$, time complexity of preparing and computing the set $A_t^r \varphi A_{t'}^r$ for all tasks and resources would be $O((C|T||R|)^2)$ in the worst case. Therefore, the time complexity of the algorithm would be $O(2|R| + \frac{|R|}{2}(C|T||R|)^2)$. The $2|R|$ term is computed for $\overline{wl}$ and LWR values (lines 1 and 2), the $(C|T||R|)^2$ and $\frac{|R|}{2}$ factors are respectively to search for the maximum similarity distance between two activities of $A_t^r \varphi A_{t'}^r$ (loop in line 9) for each LWR element (loop in line 4).

The proposed method is an extension to resource allocation algorithm and periodically rebalances the work loads of all resources. Rebalancing requires redispatching the work items from worklists of resources with workload higher than the average workload to worklists of resources with workload lower than the average workload. To be exact, for each resource r with workload lower than the average workload, two work items with maximum value $\mu^r(t') + \mu^r(t) - \delta^r(t',t)$ are chosen to be redispatched. The chosen pair of work items is executionally concurrent and would help execute faster by allocated resources. Figure 3 depicts general procedure of the proposed algorithm after allocating tasks to resources by base algorithm (like workload balancing).

4. EXPERIMENT AND DISCUSSION

To analyze the effectiveness of the proposed method, we consider a real life log. BPI challenge 2012[2] is the real life log chosen to be the benchmark dataset.

[2] http://data.4tu.nl/repository/uuid:3926db30-f712-4394-aebc-75976070e91f

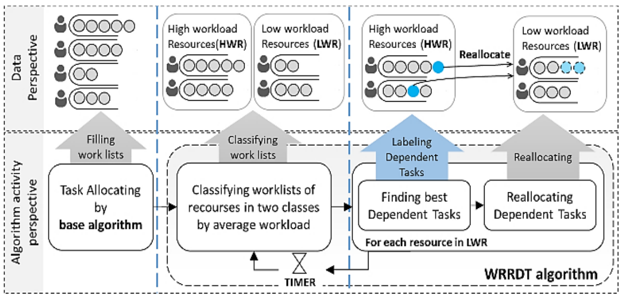
Data Perspective
Algorithm activity perspective
High workload Resources(HWR)
Low workload Resources (LWR)
High workload Resources (HWR)
Reallocate
Low workload Resources (LWR)
Filling work lists
Classifying work lists
Labeling Dependent Tasks
Reallocating
Task Allocating by base algorithm
Classifying worklists of recourses in two classes by average workload
Finding best Dependent Tasks
Reallocating Dependent Tasks
For each resource in LWR
TIMER
WRRDT algorithm

Figure 3. Graphical model of the proposed algorithm

The event log of a real-life process (see Figure 4) of BPI Challenge 2012, adopted from a Dutch financial institute, is used for simulation. Of 13078 cases, 5 activities and 55 resources are acquired from the real-life log.

In this section, the proposed method will be evaluated with some algorithms presented in other researches. These algorithms are listed in Table 1. We use 200 customer requests with different arrival rates (between 1/15 to 1/3) for the 6 methods listed in Table 1. To ensure the resulting values, experiments are repeated 10 times for each method and we used mean values to depict in Figure 5.

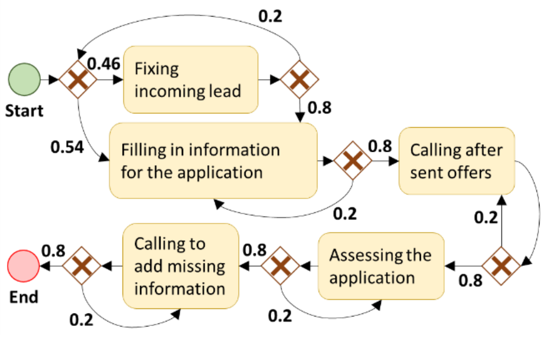
0.2
0.46
Fixing incoming lead
0.8
Start
0.54
Filling in information for the application
0.8
Calling after sent offers
0.2
0.2
0.8
End
Calling to add missing information
0.8
Assessing the application
0.8
0.2
0.2

Figure 4. BPMN[1] model for real-life financial process [25]

TABLE 1. List of some task allocation algorithms

Short name	Description (publication year)	Ref.
DSWB	Dynamic Stochastic task allocation approach for Workload Balancing as a base allocation (2015)	[26]
NBSR	Naïve Bayes Selection Rule (2015)	[24]
QL+SR	Q-learning for task allocation based on Social Relation (2015)	[33]
WB	Workload Balancing task allocation with queueing network model (2004)	[22]
WB+DR	Dynamic Rebalancing task allocation approach for Workload Balancing as a base allocation (2006)	[25]
WB+WRRDT	Workload rebalancing by reallocating dependent tasks for Workload Balancing as a base allocation (the proposed method)	

The results of comparison between 6 fore-mentioned on 'financial' process are obtained in Figures 5. It shows the proposed algorithm by using the *concurrent execution of dependent tasks* of reduced cycle time of the process more than other state-of-the-art methods.

Figure 6 depicts minimum and maximum value of resource workloads for each algorithms mentioned in Table 1. If the difference between minimum and maximum value is smaller, the dispatching of tasks between resources is more balance. In Figure 6, the workload balancing (WB) algorithm has best balance dispatching of tasks. This algorithm allocate tasks base on queueing network factors then it creates optimal workload balancing for resources. After WB algorithm, the WB+WRRDT, DSWD and WB+DR create better workload balancing for resources, respectively. These algorithms use reallocating technique at business process runtime after allocating WB algorithm as base algorithm.

At the runtime of the business process execution (not during the execution of the work item), a set of the work items in the worklist of resources are placed by allocation algorithm. Some of these work items in the worklist wait for the end of serving the resources. The complementary algorithm can reallocate pending work items from the resource worklist to another, in other words, it moves work items between worklists at the process instance runtime.

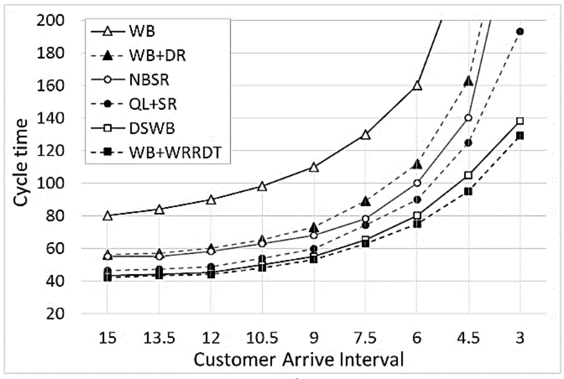

Figure 5. Comparison of the cycle time results

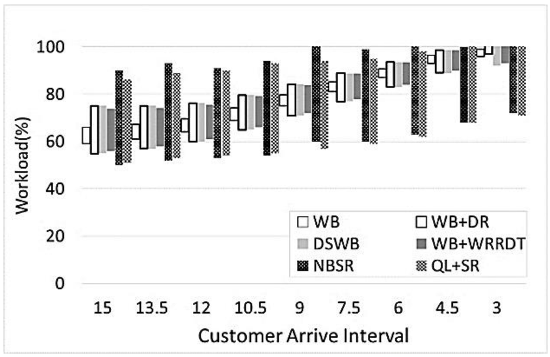

Figure 6. Comparison of the workload percentage range

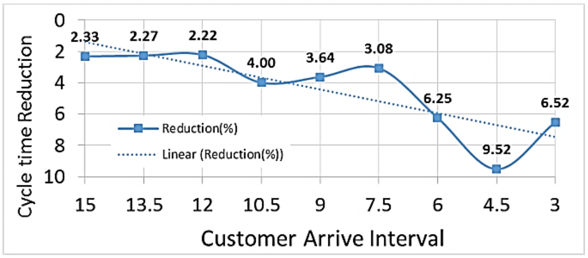

Figure 7. The cycle time reduction percentage rate of the proposed method (WB+WRRDT) in contrast to the DSWB method

This technique causes to more reduction in cycle time but a little away from the optimal workload balancing.

Figures 5 and 6 show the proposed algorithm in both its objectives of workload balancing and cycle time reduction provides better results in contrast to others.

The cycle time reduction percentage rate of the proposed method (WB+WRRDT) in contrast to the DSWB method for different arrival rates (between 1/15 to 1/3) are shown in Figure 7. The maximum reduction with value 9.52% is happened in arrival rate of 1/4.5 and the minimum reduction with value 2.22% is happened in arrival rate of 1/12. The average cycle time reduction of the proposed algorithm in contrast to the DSWB algorithm is 4.42%. The linear trend of reduction increases by increasing arrival rate. When the arrival rate increases, then the number of work items in work list of resources and the probability of dependent tasks reallocating is increased. In addition, the more reduction in cycle time is occurred.

5. CONCLUSION

In this paper, workload balancing and business process cycle time reduction are the two main objectives. By analyzing the real-life dataset of BPI, Challenge 2012 showed that there is a dependency in concurrent execution of tasks which causes that their concurrent execution have less time to separate execution of them. Therefore, we used this property which was implied in the event log and introduced an algorithm to use the property in allocation of tasks to resources and dependent tasks to be used in work list of resources. Therefore, the flowtime of case in business process reduces in addition to balancing the workload of resources. The proposed method was compared with other algorithms and the expected results were obtained.

The proposed algorithm has a considerable advantage compared to the other algorithms. First, it does not create adverse effects in workload balancing of resources. The second advantage is more consistent to increase customer arrival rate, because the probability of synchronizing the dependent tasks increases by increasing the customer arrival rate. Third, the proposed algorithm reduces the cycle time of process 4.42% more than the other algorithms. Finally, there is the possibility of combining it with other algorithms in researches.

6. REFERENCES

1. Hammer, M., "The agenda: What every business must do to dominate the decade, Crown Pub, (2003).
2. Hammer, M. and Champy, J., "Reengineering the corporation: Manifesto for business revolution, a, Zondervan, (2009).
3. Smith, H. and Fingar, P., "Business process management: The third wave, Meghan-Kiffer Press Tampa, Vol. 1, (2003).
4. Van Der Aalst, W.M., Ter Hofstede, A.H. and Weske, M., "Business process management: A survey", in International conference on business process management, Springer., (2003), 1-12.
5. Chang, D.-H., Son, J.H. and Kim, M.H., "Critical path identification in the context of a workflow", ***Information and Software Technology***, Vol. 44, No. 7, (2002), 405-417.
6. Reijers, H.A., "Design and control of workflow processes: Business process management for the service industry, Springer-Verlag, (2003).
7. Narahari, Y., Viswanadham, N. and Kumar, V.K., "Lead time modeling and acceleration of product design and development", ***IEEE Transactions on Robotics and Automation***, Vol. 15, No. 5, (1999), 882-896.
8. Zerguini, L. and van Hee, K.M., "A new reduction method for the analysis of large workflow models", in Promise, Citeseer., (2002), 188-201.
9. Son, J.H. and Kim, M.H., "Improving the performance of time-constrained workflow processing", ***Journal of Systems and Software***, Vol. 58, No. 3, (2001), 211-219.
10. Huang, Z., van der Aalst, W.M., Lu, X. and Duan, H., "Reinforcement learning based resource allocation in business process management", ***Data & Knowledge Engineering***, Vol. 70, No. 1, (2011), 127-145.
11. Van Der Aalst, W.M., Van Hee, K.M. and Reijers, H.A., "Analysis of discrete-time stochastic petri nets", ***Statistica Neerlandica***, Vol. 54, No. 2, (2000), 237-255.
12. van Hee, K. and Reijers, H., "An analytical method for computing throughput times in stochastic workflow nets", ***Simulation in Industry***, Vol. 10, No. 3, (1999).
13. Kumar, P. and Meyn, S.P., "Stability of queueing networks and scheduling policies", ***IEEE Transactions on Automatic Control***, Vol. 40, No. 2, (1995), 251-260.
14. Culler, D.E., Singh, J.P. and Gupta, A., "Parallel computer architecture: A hardware/software approach, Gulf Professional Publishing, (1999).
15. Grosu, D. and Chronopoulos, A.T., "Algorithmic mechanism design for load balancing in distributed systems", ***Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on***, Vol. 34, No. 1, (2004), 77-84.
16. Rahm, E., "Dynamic load balancing in parallel database systems", in Euro-Par'96 Parallel Processing, Springer., (1996), 37-52.
17. Xu, J., Liu, C. and Zhao, X., "Resource allocation vs. Business process improvement: How they impact on each other", in BPM, Springer. Vol. 2008, (2008), 228-243.
18. Huang, Z., Lu, X. and Duan, H., "A task operation model for resource allocation optimization in business process management", ***Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on***, Vol. 42, No. 5, (2012), 1256-1270.
19. Jin, L.-j., Casati, F., Sayal, M. and Shan, M.-C., "Load balancing in distributed workflow management system", in Proceedings of the 2001 ACM symposium on Applied computing, ACM., (2001), 522-530.
20. Zhao, W., Yang, L., Liu, H. and Wu, R., The optimization of resource allocation based on process mining, in Advanced intelligent computing theories and applications. 2015, Springer. 341-353.
21. Zur Muehlen, M., "Organizational management in workflow applications–issues and perspectives", ***Information Technology and Management***, Vol. 5, No. 3-4, (2004), 271-291.
22. Ha, B.-H., Bae, J. and Kang, S.-H., Workload balancing on agents for business process efficiency based on stochastic model, in Business process management. 2004, Springer.195-210.
23. Nisafani, A.S., Wibisono, A., Kim, S. and Bae, H., "Bayesian selection rule for human-resource selection in business process management systems", ***Journal of Society for e-Business Studies***, Vol. 17, No. 1, (2014).
24. Wibisono, A., Nisafani, A.S., Bae, H. and Park, Y.-J., On-the-fly performance-aware human resource allocation in the business process management systems environment using naïve bayes, in Asia pacific business process management. 2015, Springer.70-80.
25. Ha, B.-H., Bae, J., Park, Y.T. and Kang, S.-H., "Development of process execution rules for workload balancing on agents", ***Data & Knowledge Engineering***, Vol. 56, No. 1, (2006), 64-84.
26. Xie, Y., Chien, C.-F. and Tang, R.-Z., "A dynamic task assignment approach based on individual worklists for minimizing the cycle time of business processes", ***Computers & Industrial Engineering***, Vol. 99, No. 1, (2015), 401-414.
27. Xie, Y., Chien, C.-F. and Tang, R.-Z., "A method for estimating the cycle time of business processes with many-to-many relationships among the resources and activities based on individual worklists", ***Computers & Industrial Engineering***, Vol. 65, No. 2, (2013), 194-206.
28. Coffman, E.G. and Bruno, J.L., "Computer and job-shop scheduling theory, John Wiley & Sons, (1976).
29. Baker, K.R., "Introduction to sequencing and scheduling, John Wiley & Sons, (1974).
30. Pinedo, M., *Scheduling: Theory, algorithms and systems, 1995*, Prentice-Hall, Englewood Cliffs, NJ.
31. Rhee, S., Bae, H., Ahn, D. and Seo, Y., "Efficient workflow management through the introduction of toc concepts", in Proceedings of the 8th annual international conference on industrial engineering theory, applications and practice (IJIE2003)., (2003).
32. Shen, M., Tzeng, G.-H. and Liu, D.-R., "Multi-criteria task assignment in workflow management systems", in System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on, IEEE., (2003), 9-17.
33. Liu, X., Chen, J., Ji, Y. and Yu, Y., "Q-learning algorithm for task allocation based on social relation", ***Process-Aware Systems***, (2015), 49-58.
34. Kumar, A., van der Aalst, W.M. and Verbeek, E.M., "Dynamic work distribution in workflow management systems: How to balance quality and performance", ***Journal of Management Information Systems***, Vol. 18, No. 3, (2002), 157-193.
35. Larbi, S. and Mohamed, S., "Modeling the scheduling problem of identical parallel machines with load balancing by time petri nets", ***International Journal of Intelligent Systems and Applications (IJISA)***, Vol. 7, No. 1, (2014), 42-48.

36. Ly, L.T., Rinderle, S., Dadam, P. and Reichert, M., "Mining staff assignment rules from event-based data", in Business process management workshops, Springer., (2006), 177-190.

37. Yang, H., Wang, C., Liu, Y. and Wang, J., "An optimal approach for workflow staff assignment based on hidden markov models", in On the Move to Meaningful Internet Systems: OTM 2008 Workshops, Springer., (2008), 24-26.

38. Eder, J., Pichler, H., Gruber, W. and Ninaus, M., "Personal schedules for workflow systems", in International Conference on Business Process Management, Springer., (2003), 216-231.

39. Motameni, H., Movaghar, A. and Amiri, M.F., "Mapping activity diagram to petri net: Application of markov theory for analyzing non-functional parameters", ***International Journal of Engineering Transactions B Applications***, Vol. 20, No. 1, (2007), 65-73.

Cycle Time Reduction and Runtime Rebalancing by Reallocating Dependent Tasks

M. Yaghoubi*[a,b], M. Zahedi[b]

[a] *Computer Engineering Department, Faculty of Engineering, Golestan University, Iran*
[b] *Computer & IT Engineering Department, Shahrood University of Technology, Shahrood, Iran*

PAPER INFO

Paper history:
Received 24 April 2017
Received in revised form 09 July 2017
Accepted 08 September 2017

Keywords:
Business Process
Cycle Time
Workload Balancing
Resource Allocation
Dependent Tasks
Task Concurrency

چکیده

سیستم های مدیریت فرآیند کسب و کار (BPMS) یک سیستم اطلاعات پیچیده است که طراحی، مدیریت و بهبود فرآیندهای کسب و کار را فراهم می کند. تخصیص وظیفه به منابع انسانی یکی از مهمترین مسائلی است که باید در BPMSمدیریت شود. الگوریتم تخصیص وظیفه به منظور رفع سیاست های مختلف سازمان ها تعریف شده است. مهمترین این سیاستها می تواند کاهش زمان چرخه متوسط، متعادل نمودن حجم کار منابع، افزایش کیفیت محصول و به حداقل رساندن هزینه های تولید باشد. بنابراین، انتخاب یک منبع مناسب در الگوریتم های تخصیص وظایف می تواند بر کل سیاست سازمان تأثیر بگذارد. در شرایط بار سنگین و یا زمانی که تعداد منابع انسانی محدود است، تعادل کار در سیستم می تواند ثبات سیستم را افزایش دهد. در این مقاله یک الگوریتم تخصیص وظیفه برای تعادل بار کاری در زمان اجرا ارائه می شود و حداقل زمان چرخه را با ارائه یارانه های وابسته به منابع برای پردازش همزمان فراهم می کند. نتایج تجربی نشان می دهد که ترکیبی از الگوریتم های قبلی با الگوریتم های پیشنهادی، در مقایسه با اکثر الگوریتم های پیشرفته ترین حالت، باعث کاهش ٪ ۴/۴۲ در زمان چرخه می شود.

doi*: 10.5829/ije.2017.30.12c.03*