A Temporal Logic Based Grid Workflow Model and Scheduling Scheme*

Yu-E Zheng, Huadong Ma, Lejun Zhang
Beijing Key Lab of Intelligent Telecommunications Software and Multimedia,
Beijing University of Posts and Telecomm., Beijing 100876, China
ilovezye@yahoo.com.cn

Abstract

Recently, grid computing has become a popular technology for large-scale scientific or business computing. Grid workflow system can enhance the flexibility and efficiency of the grid computing. By the virtue of the dynamic and heterogeneous nature of grid environment, grid workflow system requires a resource evaluation and utilization enhancing mechanism to satisfy all kinds of grid resource requirements from workflow instances and ease to build billing system.

Motivated by this, we present a novel Extended Temporal Logic based Workflow Specification (ETLWS) model to meet with the requirements above. The ETLWS extends temporal logic based workflow specification model by injecting resource requirement descriptor. Then, we propose a dynamic workflow scheduling scheme to improve the stability and efficiency of grid workflow system. In particular, we implement a system prototype and present a case study. The experimental results demonstrate the effectiveness of the ETLWS model and its corresponding scheduling scheme.

1. Introduction

Grid workflow system is supposed to support modeling and execution of large-scale sophisticated e-science and e-business processes in grid environment. This system can utilize heterogeneous and distributed grid resources (e.g. computing resource, storage resource, and data resource) to implement sophisticated application flows. Potential applications span a wide spectrum: biometrics, business oriented data-mining and multimedia oriented computing. In such large-scale sophisticated computing applications, grid workflow provides a mechanism which is concerned with the automation of procedures and the efficiency. Therefore, many international organizations and research institutions drafted and constituted a series of standards and workflow infrastructures, such as Grid Flow Service [1] proposed by OGSA and Grid Service Flow Language [2] proposed by S. Krishnan. However, workflow system in grid environment has two special characteristics. First, grid environment refers to the Internet-connected computing environment in which computing and data resources are geographically dispersed in different administrative domains. Second, it is necessary to ensure the required resources for the workflow instance in such heterogeneous environment. Therefore, how to establish a resource evaluation and utilization enhancing mechanism to improve the stability and efficiency of grid workflow system is a challenging topic.

Motivated by the problems mentioned above, we combine the temporal logic based workflow specification model (TLWS) [3] with resource requirement descriptor (RRD). RRD is a group of expressions which describes the grid resource requirement of an activity in a workflow instance. For example, the number of CPUs for computing, the free disk space for saving temporary files, or the requirement of some kind of related software to support an application. Due to the introduction of RRD, a grid workflow scheduler can dynamically and reasonably assign grid resources to the corresponding tasks in the workflow instance. Meanwhile, compared with other workflow specification models, such as DAG [4], Petri-Net [5], the TLWS model is more flexible to construct workflow. Hence we inject RRD into the temporal logic based workflow model, and thus propose an improved model named as Extended TLWS (ETLWS). In particular, we propose an auction mechanism based and priority-driven scheduling (APS) scheme to enhance the stability and efficiency of grid workflow.

* The work is supported by the National Natural Science Foundation of China (90612013), the National High Technology Research and Development Program of China under Grant No.006AA01Z304, the Specialized Research Fund for the Doctoral Program of Higher Education (20050013010) and the NCET Program of MOE, China.
The rest of this paper is organized as follows. Section 2 reviews the related works. In Section 3, we describe the resource requirement descriptor (RRD) and ETLWS model. Section 4 presents a workflow scheduling schema APS. Through a typical case, we detail the implementation of our ETLWS model in Section 5. Finally, Section 6 concludes the paper and outlines the future work.

2. Related Works

The grid workflow system plays an increasingly important role in grid computing. Many researchers and organizations have made great contributions to grid workflow system. The existing works can be classified as follows:

Workflow models: the main function of workflow model is the specification of temporal synchronization among activities. We categorize the existing workflow models as follows:
1) Graph model [4]. A workflow is defined as a directed acyclic graph with a beginning node and an ending node;
2) Petri-Net model [5]. Place and transition in Petri-Net are used in specifying workflow service process, where a place represents a control node and a transition represents an activity. A specific Petri-Net model for workflow is WorkflowNet (WF-Net);
3) Event-Condition-Action (ECA) model [6]. An ECA rule is used to specify service process.
4) CTR model [7]. Davulcu et al. used Concurrent Transaction (CTR) Logic to specify and analyze the workflow service process precisely;
5) Temporal Logic based Workflow Specification model (TLWS) [3]. The temporal logic based XYZ/E system is used to describe the workflow model.

Grid Workflow Infrastructure Drafts and Specifications: Some academic and industrial organizations have launched the pioneer researches. For instance, GSFL [2] has specified a Web-Service based infrastructure which is OGSA-compatible. Bivens et al. have proposed Grid Workflow [8] which focuses on data computing and transmitting. The resource query [8], which is similar to RRD, is also proposed in this paper. However, the drawback of the Grid Workflow in [8] is that the ability of describing logic process is limited. Gridant [9] is a powerful grid workflow tools built based on Apache Ant [10] and Globus Tookit [11]. Gridant is convenient to describe a grid workflow by XML. However, it is weak for Gridant in describing required grid resource and logic process.

Grid Toolkit: With the development of grid drafts and specifications, many grid middleware and tools are emerging. Globus Toolkit 4 [11], based on the Web Service technology and WSRF, is compatible with OGSA standards. Globus is a popular grid standard, and our workflow prototype is also based on it.

Resource scheduling and allocation algorithm: the algorithm compares resources in grid environment to the commodity in the market. Using the equilibrium theory in economics, the algorithm adjusts resource price according to the demand and supply of resource for achieving the goal of high efficient utilization of resources in response to current resource prices. In [12] Zhang et al. present a grid resource allocation and management model based on micro-economic theory; and In [13], Weng et al. present a resource price determination algorithm with the economic mechanism. Other resource allocation algorithm is based on price adjustment algorithm [12].

However, the specification models mentioned above haven’t paid more attention to the heterogeneity of grid environment as well as the grid resource requirement description and resource utilization. In this paper, we are making an effort to fill this gap.

3. Resource Requirement Descriptor and ETLWS model

3.1. Resource Requirement Descriptor

Resource requirement descriptor (RRD) is a group of expressions describing grid resources required by an activity. For example, a data-mining workflow is composed of four activities, and one of these activities needs 10 CPUs, 300M free disk space and GMPI [12] component to support its application. So a submitter can describe his requirement using RRD in the workflow description file in order that the workflow scheduler can assign the appropriate grid resource to the relevant activity.

The major advantages of RRD are as follows:
• Meet with the resource requirement of the specific computing task. As mentioned above, most scientific or business applications require the specific grid resource. The grid workflow system could assign the specific grid resource to the application; otherwise the application could not be executed successfully.

2 GMPI (Grid Message Passing Interface) is a software component we developed for supporting the communication between the parallel processes in grid environment.
• Ensure that the workflow instance can subscribe all the required resources before the running of activity to ensure the stability of the workflow instances and reduce the execution time.
• Ease to build grid resource billing system. Based on RRD, the billing system is easy to account the charge of a workflow instance.

3.2. ETLWS model

TLWS model is based on XYZ/E system [14]. Compared with other workflow models, TLWS model can support step-wise refinement design and verification on fast rapid prototyping [15]. Due to these advantages, we propose an extended TLWS model as a grid workflow specification model. The main differences between ETLWS model and TLWS model are the RRD and its operator.

Definition 3.1. TLWS model is defined as a six tuple [3], $TLWS = (\Sigma, F, R, T, O, S_0)$, where $\Sigma$ is a set of scene states; $R$ is a set of roles; $T$ is a set of tasks; $F$ is called as a set of scene state transition rules, and a rule is formed as $lb = s \rightarrow P \rightarrow \neg (Q \land lb = S) \land \neg \neg (Q \land lb = S)$ where $lb$ is state control variable, $@i$ is temporal logic operator, and $Q_i$ denotes the operation if $Pi$ is true; $O$ is a mapping from rule to activity expression:

- $F \rightarrow \{\text{activity expression}\} \cup \{\omega\}$;
- $S_0$ is the initial state of the workflow.

Definition 3.2. Activity operator [3]. If $A$, $B$ and $C$ are activities, the activity operators between activities are defined as follows:

- Sequence: $A:B$. $A$ is presented first then $B$. The expression ends when $B$ ends.
- AND: $A \land B$. $A$, $B$ start simultaneously, and the expression ends when both of activities end.
- OR: $A \lor B$. $A$ or $B$ starts, and the expression ends when either of activities ends.
- Case: $(case_1 \land A) \lor (case_2 \land A) \lor \ldots \lor (case_n \land A)$. If $case_i$ is True, the activity $i$ is executed ($1 \leq i \leq n$).
- Loop: $A^*m$. Repeat $m$ times to present $A$.

Here, we give the extended definitions.

Definition 3.3. Grid Resource Set $R$. $R$ is denoted as $R = \bigcup_{i=1}^{n} R_i$, where $R_i$ denotes a grid resource (e.g. CPU, storage space) and $R_i \cap R_j = \emptyset$ ($1 \leq i, j \leq m$). Further, $R_i$ can be denoted as $R_i = (I_i, A_i, D_i)$ where $I_i$ is a resource identifier, $A_i$ is a set of attributes, and $D_i$ is a set of descriptors which describes the values of corresponding items in $A_i$.

Definition 3.4. RRD is a two tuple denoted as $RRD=(A, D)$ where $A$ is attribute set and $D$ is the value set of the corresponding attributes in $A$.

Definition 3.5. Atomic Activity. Atomic Activity can be defined as a four tuple: $A = (Participant, Application, Relevant_Data, RRD_Set)$ where

- Participant is a set of participants in an activity. The element of the Participant might be an organization, a role, a person or a system.
- Application is a set of applications in an activity. These applications are invoked by the workflow process in order to support the activity or complete the activity automatically.
- Relevant Data is a set of relevant data in an activity. Relevant Data includes output data produced by the activity and input data necessary to the activity.

- $RRD_Set$. $RRD_Set = \bigcup_{i=1}^{m} RRD_i$, where $RRD_i$ is a resource requirement descriptor. $RRD_Set$ is the grid resource requirement of the activity.

According to Definition 3.2, we propose Definition 3.6 to point out what the workflow should do when the existing grid resources cannot satisfy the requirement described by RRD.

Definition 3.6. RRD operator. Assume that $A$, $B$ are activities, $d$ is a constant to denote duration; activity $B$ is a special activity to denote the end of the workflow. $R$ is a set of grid resources. $R = \bigcup_{i=1}^{n} R_i$, $R_i = (I_i, A_i, D_i)$. $RRDS$ is a set of resource requirement descriptors. $RRD$ is an element in the $RRDS$. $Match(RRDS, R)$ is a function of matching, and it is true if and only if for any $1 \leq i \leq n$, $1 \leq j \leq m$, $\exists RRD_i \in RRDS$ s.t. $A_r = (I_r, A_r, D_r)$, when $A_i \in A_r$, there is corresponding $D_r \subset D_i$ i.e. there are matching grid resources satisfying the requirement described by $RRD_i$. Three RRD operators are defined as follows:

1) $\text{WAIT}(A, d) = ((\forall Match(RRDS, R) \land m) \lor \forall (Match(RRDS, R) \land (A \land d=0)))*d$.

If there is no matching grid resource, then wait for the resource in the duration $d$. $A$ will be executed when the resource is discovered or released;

2) $\text{BREAK} (A) = \neg Match(RRDS, R) \lor B$ $\lor \forall Match(RRDS, R) \land \neg (A \land d=0))$.

If no matching grid resource, break the workflow;

3) $\text{RR}(A) = (Match(RRDS, R) \land (A \land d=0)) \lor (\forall Match(RRDS, R) \land A)$.

If no matching grid resource, the workflow instance reduces the resource requirement.

3.3. Grid Workflow Specification Language

The Grid Workflow Specification Language (GWSL)
is based on ETLWS model and Workflow Specification Language (WSL). The differences between GWSL and WSL are: 1) the GWSL is based on XML; 2) the RRD and RRD operator are introduced in GWSL.

A GWSL instance is composed of two parts:
1) The description of the activities;
2) The description of the logic process.

The basic construction of the logic statement in GWDL is as follows [3]:
<id>^<condition expression>^$O<RRD_Operator (activity)>^$O<next id>
where <id> is the identifier of the statement; <condition expression> is the condition of activity execution; RRD_Operator() is what workflow should do when there is no matching grid resources described by RRD and <next id> is the next statement when the activity completed. For example, the statement
lb=A^(a= true)=>$O{WAIT(A,d)}^$Olb=D;
indicates that if a=true, the activity A turns into ready state, the next statement is lb=D; if there exist grid resources matching the RRD of A in the duration d, then A is executed; otherwise the entire workflow stop and exit.

A case of workflow process is illustrated in Figure 1.
We use the GWSL to specify the workflow illustrated in Figure 1. We take two steps to finish the work.
First, an activity can be described as follows

4. Priority Driven Scheduling Schema
4.1. Scheme Overview
In order to enhance the efficient utilization of resources and exert the effect of ETLWS model, we
proposed a PSP \(^3\) Auction mechanism based and Priority-driven Scheduling (APS) schema for grid workflow system. According to PSP auction mechanism \([12]\), it is supposed that the activity in workflow is a bidder in the auction who bids the grid resources. The activity (bidder) with the highest value wins the grid resources and executes its applications on the grid resources.

APS scheme consists of activity selection algorithm and resource allocation algorithm. The procedure of the algorithm is shown in Figure 2. Activity selection algorithm is in charge of selecting the activity with top priority. And resource allocation is responsible for allocating the grid resource to the selected activity. In brief, the activity selection handles which activity should be executed and the resource allocation handles where the activity should be executed.

As shown in Figure 2, we further describe the procedure of scheduling as follows:

1) The scheduler gets the activities from ready activity queue and evaluates the priority level of it (step 1 in Figure 2).
2) The scheduler picks out the activity with top priority level and calculates the bid valuation of the required resources (described in RRD) due to the resource’s priority level (step 2 and 3).
3) Activity submits the bid to the resource auctioneer (step 4). Then, resource auctioneer broadcasts the bid information in the resource market (step 5).

4) For every grid resource, it may receive many bids from the activities. According to the auction mechanism, the winner among these bidders must meet two conditions: 1) the valuation from the bidder is higher than the price of the grid resource; 2) the valuation from the bidder is the highest one. And according to PSP, the payment for the resource is the second-highest bid. Resource auctioneer would send the second-high price to the activity (step 6).
5) Similarly, for every activity (bidder), it may win numbers of resources with different prices. It would choose the resource with the lowest bid price and its application would be executed on this resource (step 7).
6) In Figure 2, A is a price adjustment process which uses the microeconomics supply and demand equilibrium theory to dynamic adjusts the price of resource with certain cycle. The price adjustment algorithm is described in \([12]\). With the price adjustment, the utilization of grid resource would be improved.

4.2. Activity Selection Algorithm

The activity selection algorithm selects the activity and evaluates its bid valuation. The selection is based on the priority level of the activity.

Assume that \(d\) is an integer which denotes the maximum priority level, \(CA\) is a set of the activities in the critical path of the workflow instances. The sum of the require time (a RRD) of the activities in the critical path is the maximum.

Let the function \(priolev(A)\) to express the priority level of \(A\). \(priolev(A)\) is an integer. The larger \(priolev(A)\) is the higher priority \(A\) is.

The rules of \(priolev(A)\) are as follows:

1) Initially, \(priolev(A) = \lfloor \frac{d}{2} \rfloor\);
2) If \(A \in CA\), i.e. the activity \(A\) is in the critical path, then \(priolev(A) +=2\);
3) If the RRD operator of \(A\) is BREAK, then \(priolev(A) +=1\);
4) If the RRD operator of \(A\) is WAIT, then \(priolev(A) -=1\);
5) If \(priolev(A) > d\), then \(priolev(A) = d\). If \(priolev(A) < 1\) then \(priolev(A) = 1\).

Let the function \(valuation(A)\) to indicate the bid. \(valuation(A) = priolev(A) \times u\) where \(u\) is the unit of the bid price.

---

\(^3\) Second-Price Sealed bid auction. In the mechanism each bidder submits a sealed bid to the seller. The high bidder wins and pays the second-highest bid for the good.
The activity selection algorithm is described in Algorithm AS(). In this algorithm, if there is no matching resource for the activity in the environment, its priority level will increase, that means the resource bid from the activity increases. As a result, it is easier for the activity with high priority level to get the matching resource. Meanwhile, the increasing priority level can prevent activity from hungry.

Algorithm AS() //Activity Selection
Begin
    Input: a queue of ready activities RQ;
    Output: the scheduling of the activities in RQ;
    d: integer; //denotes top priority level
    While true {
        Sort RQ according to PrioLev(Ai);
        For every Ai in RQ{
            If PrioLev(Ai)>d and no matching resource
                Continue;
            If there is matching resource in grid environment{
                Valuation(Ai) = PrioLev(Ai)* u;
                Send Valuation(Ai) to resource auctioneer;
            }
            Else
                PrivLev(Ai)++;
        }
    }
End

4.3. Resource Allocation Algorithm.

The price adjustment algorithm is described in [12]. The main function of the price adjustment algorithm is guaranteeing the equilibrium between the supply and the need of the grid resources to enhance the utilization of grid resource. The initial price of the grid resource is du/2. And the algorithm adjusts the price according to the demand and supply of the grid resource, if the demand is more than supply, the price will increase, and else the price will reduce.

The resource auctioneer algorithm is described in Algorithm RA() and the resource market algorithm is illustrated in Algorithm RM().

Algorithm RA() //Resource Auctioneer
Begin
    Input: a bidder set B;
    Output: the resource auction result;
    For every bidder in B {
        Broadcast bid to every matching grid resource;
        Get the resource list which the bidder wins in the auction;
        Response the resource list to the activity;
    }
End

Algorithm RM() //Resource Market
Begin
    Input: the queue of bids from auctioneer;
    Output: the resource list for all bids;
    For every resource R in resource market {
        Get the corresponding bids from auctioneer;
        For every bid B {
            If price of B > the price of R and price of B is highest in all prices {
                Get the second-highest price as the final price;
                Put R to the resource list of B;
            }
        }
    }
End

5. Case Study and Performance Evaluation

In order to perform empirical evaluation of our APS scheme, we implement a simple experimental prototype. The prototype is based on Globus Tookit 4 and developed by JAVA. The infrastructure of the experimental prototype is shown in Figure 3. In the prototype, we use the ETLWS based GWSL as the workflow description language and APS scheme as the scheduling scheme.

![Figure 3. The infrastructure of the simple prototype](image-url)
The test environment is constructed by a Dell Power-edge 2850 server on which the workflow system server-end deploys and other 20 additional computing nodes. The case is a data-mining workflow for telecommunication customers value analysis.

5.1 Case Study

In this section, we use a case to illustrate the effectiveness of the APS scheme.

The case of workflow shown in Figure 4 contains six activities. At the beginning, the activity A reads a text file of customer information as its input, and finally the activity F produces a text file for customer value evaluations. And the application of activity A, C, and D supports parallel computing. Depending on GMPI component, the program of A can execute several processes on different computing nodes in grid environment. Figure 5 shows the results of the case.

![Figure 4. The workflow of telecommunication customer value analysis. A: clique computing; B: file combination; C/D: clique number analysis; E: file combination; F: customer value evaluation.](image)

We did the experiment for comparing two scheduling scheme: ETLWS based APS scheme and FCFS scheme. Four different scales of workflow instances (15, 20, 30 and 40) are run using APS scheme and FCFS scheme, respectively. The performance of APS schema and the FCFS scheme are measured for each workflow instance scale.

![Figure 5. The results of telecommunication customers value analysis. Each node (small rectangle in Figure 4) denotes a customer and each edge denotes a communication between two customers. The more edge a customer has, the more valuable the customer is.](image)

![Figure 6. The performance comparison between APS scheme and FCFS scheme](image)

5.2. Performance Evaluation

In the case, for the APS scheme, we let the top priority \(d=10\), the unit of price \(u=2\). And according to the requireTime tag in RRD, the critical path \(A->B->C->E->F\). In order to let the activity in this path execute as quickly as possible, we make an optimal RRD Description of the case which is outlined in Table 1.

<table>
<thead>
<tr>
<th>Activity</th>
<th>DDR</th>
<th>DDR Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>&lt;minCPU&gt;4&lt;/minCPU&gt; &lt;maxCPU&gt;4&lt;/maxCPU&gt;</td>
<td>RR</td>
</tr>
<tr>
<td>B</td>
<td>&lt;minCPU&gt;1&lt;/minCPU&gt; &lt;maxCPU&gt;1&lt;/maxCPU&gt;</td>
<td>WAIT</td>
</tr>
<tr>
<td>C</td>
<td>&lt;minCPU&gt;2&lt;/minCPU&gt; &lt;maxCPU&gt;2&lt;/maxCPU&gt;</td>
<td>RR</td>
</tr>
<tr>
<td>D</td>
<td>&lt;minCPU&gt;1&lt;/minCPU&gt; &lt;maxCPU&gt;1&lt;/maxCPU&gt;</td>
<td>RR</td>
</tr>
<tr>
<td>E</td>
<td>&lt;minCPU&gt;2&lt;/minCPU&gt; &lt;maxCPU&gt;2&lt;/maxCPU&gt;</td>
<td>RR</td>
</tr>
<tr>
<td>F</td>
<td>&lt;minCPU&gt;1&lt;/minCPU&gt; &lt;maxCPU&gt;1&lt;/maxCPU&gt;</td>
<td>WAIT</td>
</tr>
<tr>
<td></td>
<td>&lt;diskSpace&gt;300&lt;/diskSpace&gt;</td>
<td></td>
</tr>
</tbody>
</table>

We did the experiment for comparing two scheduling scheme: ETLWS based APS scheme and FCFS scheme. Four different scales of workflow instances (15, 20, 30 and 40) are run using APS scheme and FCFS scheme, respectively. The performance of APS schema and the FCFS scheme are measured for each workflow instance scale.
Figure 6 shows the experimental results of workflow instance scale. From Figure 6 (a), we can see that the average CPU usage of APS scheme (71.55%) is superior to that of FCFS algorithm (59.35%) at the expense of load. In addition, in Figure 6 (b), we can find that the time cost will reduce especially when the grid workflow system is overloaded. As shown in Figure 6 (b), when 15 workflow instances are running on the system, the time cost of APS and FCFS are similar, however when 40 workflow instances are running on the system, the time cost of APS is (6356s) less than FCFS (7532s), the difference is 1176 s. In brief, with the increase of workflow scale, our model can significantly enhance the efficiency and stability of the system.

5.3 Grid Workflow Comparision

Compared with the Grid Workflow and Gridant, ETLWS prototype has some excellent features. That is, ETLWS prototype supports the RRD and RRD operators which handle the situation which is no matching grid resource and with the MPI component, the prototype supports multiple process communication between different nodes. We compare it with some other grid workflow systems in Table 2.

<table>
<thead>
<tr>
<th>Table 2. Grid Workflow Comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workflows</td>
</tr>
<tr>
<td>Required Resource Description</td>
</tr>
<tr>
<td>Required Resource Operator</td>
</tr>
<tr>
<td>Multiple process communication</td>
</tr>
<tr>
<td>Refinement design [3]</td>
</tr>
</tbody>
</table>

6. Conclusions and Future Work

In this paper, our major contribution is proposing an extended temporal logic workflow specification model with an auction mechanism based and priority-driven workflow scheduling scheme. The experimental results show that ETLWS model and its prototype can significantly enhance the stability and efficiency of the workflow system.

Considering the complexity of the temporal logic operators, the operators are not completely implemented in the model. Moreover, the scheduling scheme still needs to be improved. We will strengthen them in our future work.

References