

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

A collaborative scheduling approach for service-driven scientific workflow execution

Wanchun Dou^{a,b,*}, J. Leon Zhao^c, Shaokun Fan^d^a State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China^b Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China^c Department of Information Systems, City University of Hong Kong, Kowloon, Hong Kong SAR, China^d MIS Department, University of Arizona, Tucson, AZ 85721, USA

ARTICLE INFO

Article history:

Received 13 January 2009

Received in revised form 19 August 2009

Available online 22 November 2009

Keywords:

Scientific workflow

Scheduling strategy

Private workflow fragment

Scientific collaboration

ABSTRACT

Scientific workflow execution often spans multiple self-managing administrative domains to obtain specific processing capabilities. Existing (global) analysis techniques tend to mandate every domain-specific application to unveil all private behaviors for scientific collaboration. In practice, it is infeasible for a domain-specific application to disclose its process details (as a private workflow fragment) for privacy or security reasons. Consequently, it is a challenging endeavor to coordinate scientific workflows and its distributed domain-specific applications. To address this problem, we propose a collaborative scheduling approach that can deal with temporal dependencies between a scientific workflow and a private workflow fragment. Under this collaborative scheduling approach, a private workflow fragment could maintain the temporal consistency with a scientific workflow in resource sharing and task enactments. Further, an evaluation is also presented to demonstrate the proposed approach for coordinating multiple scientific workflow executions in a concurrent environment.

© 2009 Elsevier Inc. All rights reserved.

1. Introduction

Recently, scientific workflows are gaining more and more momentums due to their key role in e-science and cyber-infrastructure applications. It is a new special type of workflow that often underlies many large-scale complex e-science applications such as climate modelling, structural biology and chemistry, medical surgery or disaster recovery simulation [1–3]. Moreover, in the past few years, some computing infrastructures, e.g., grid infrastructure, have been emerged for accommodating powerful computing and resource sharing capabilities required by scientific workflows [4,5].

Compared with business workflows, scientific workflows have special features such as computation, data or transaction intensity, less human interactions, and a larger number of activities [1]. As scientific workflows are typically data-centric and dataflow-oriented “analysis pipelines” [1,6], scientists often need to “glue” together various cross-domain services such as cross-organizational data management, analysis, simulation, and visualization services [7,8]. Accordingly, scientific workflow applications frequently require collaborative patterns marked by multiple domain-specific applications from different organizations. An engaged domain-specific application often contributes a definite local computing goal to global scientific workflow execution. In their loose coupled application environment, goal-specific scientists typically are rather individualistic and more likely to create their own “knowledge discovery workflow” by taking advantage of available services [1]

* Corresponding author at: State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China.

E-mail address: douwc@nju.edu.cn (W. Dou).

(in a grid computing infrastructure, these services for scientific collaboration are often called grid services [9]). It promotes scientific collaboration in form of service-oriented computing for achieving certain computing goals.

To facilitate service-oriented scientific workflow's development and execution, cross-domain workflow modelling and scheduling are key topics that currently cause more and more attentions [4,7,10,11]. For example, Yu and Buyya [10] provided a general taxonomy of scientific workflow, in which workflow design, workflow scheduling, fault tolerance, and data movement are four key features associated with the development and execution of a scientific workflow management system in grid environment. Furthermore, they believe that scientific workflow paradigm could greatly enhance scientific collaboration through spanning multiple administrative domains to obtain specific processing capabilities. Here, scientific collaborations are often navigated by data-dependency and temporal-dependency relations among goal-specific domain applications, in which a domain-specific application is often implemented as a local workflow fragment deployed inside a self-managing organization for providing the demanded services in time.

In this situation, effective collaborative scheduling between a scientific workflow and engaged self-managing organizations may be greatly helpful for promoting the interactions of independent local applications with the higher-level global application, through which coordinated executions could be readily available for computation- and data-rich scientific collaboration. For example, resource management in grid environment is typically subject to individual access, accounting, priority, and security policies of the resource owner. Resource sharing is, necessarily, highly controlled, with resource providers and consumers defining clearly and carefully just what is shared, who is allowed to share, and the conditions under which sharing occurs. The usage policy imposing on these resources is often enforced by a self-managing organization [9,12]. At runtime, if a self-managing organization refuses to disclose its process details for privacy or security reasons, the resource service process is often promoted by a resource-broker [13,14]. Besides, if a resource could not be shared by different resource users at the same time, executions of different scientific workflow around these resources should coordinate their resource sharing in a compromising way. Otherwise, some conflicts would be occurred during the execution. Therefore, cross-organizational scientific workflow execution, resource allocation, and compromising usage policy should be scheduled in an incorporated way in a concurrent environment [7,15].

For example, a computing center is a typical self-managing organization that often bears up heavy computing loads from numerous goal-specific applications. The scheduling of a computing center for satisfying its multi external service requirements is a typical coordinative process between a scientific workflow and a self-managing organization. Resource compromising usage policy is often recruited for coordinating its computational resource's using processes engaged in different scientific collaborations in a concurrent environment. Additionally, for a performance-driven scientific workflow execution, it is a more complex situation that the collaborative scheduling process not only covers cross-organizational resource sharing, but also covers task enactment deployed inside a self-managing organization [7,12].

Existing (global) analysis techniques often mandate every domain-specific application to unveil all individual behaviors for scientific collaboration [16]. Unfortunately, such an analysis is infeasible when a domain-specific application refuses to disclose its process details for privacy or security reasons [17,18]. Therefore, it is always a challenging endeavor to coordinate a scientific workflow and its distributed domain-specific applications (private workflow fragments), especially when a local workflow fragment is engaged in different scientific workflow executions in a concurrent environment.

In view of these observations, a collaborative scheduling approach is investigated. In this paper, for achieving coordinated executions of a scientific workflow. Taking advantage of the collaborative scheduling strategy, a private workflow fragment could maintain its temporal consistency with a scientific workflow in resource sharing and task enactments. *Please note that our method subscribes to relative time rather than absolute time in collaborative scheduling applications.* The rest of this paper is organized as follows. In Section 2, the application context and temporal context of a cross-domain scientific workflow are investigated. In Section 3, a temporal reasoning rule is put forward for collaboration scheduling application of a scientific workflow. In Section 4, an evaluation is proposed for demonstrating our approach in coordinating multiple scientific workflow executions in a concurrent environment. In Section 5, related works and comparison analysis are presented to evaluate the feasibility of our proposal. Finally, the conclusions and our future work are presented in Section 6.

2. Context analyses of scientific workflow execution

As mentioned in [10], cross-organizational collaboration engaged in a scientific workflow often aims at obtaining specific processing capabilities through spanning multiple administrative domains. Here, a domain-specific application engaged in scientific collaboration is often uniquely associated with a local workflow fragment deployed in a self-managing organization. In this paper, when a workflow fragment refuses to disclose some of its process details for privacy or security reasons, it would be treated as a private workflow fragment of a self-managing organization. For a private workflow fragment, the actions and resources hidden from a scientific workflow specification and execution would be treated as **silent** actions and **silent** resources.

Compared with the **silent** actions and **silent** resource, a self-managing organization only exposes its publicly accessible **port** for its scientific collaboration. Therefore, a private goal-specific workflow fragment consists of a set of silent actions, silent resources and some publicly accessible ports. It is essentially a gray box embedded in a scientific workflow. In scientific workflow execution, it is wholly a functional unit for scientific collaboration, and is triggered by its publicly accessible port for certain computing goals. In this paper, a publicly accessible port would be treated as an interaction interface between a scientific workflow and a self-managing organization.

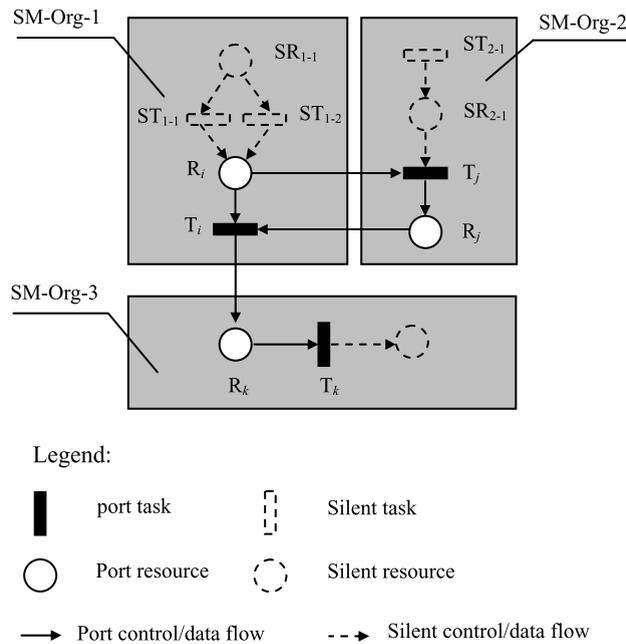


Fig. 1. Global application context of a cross-organizational scientific workflow execution.

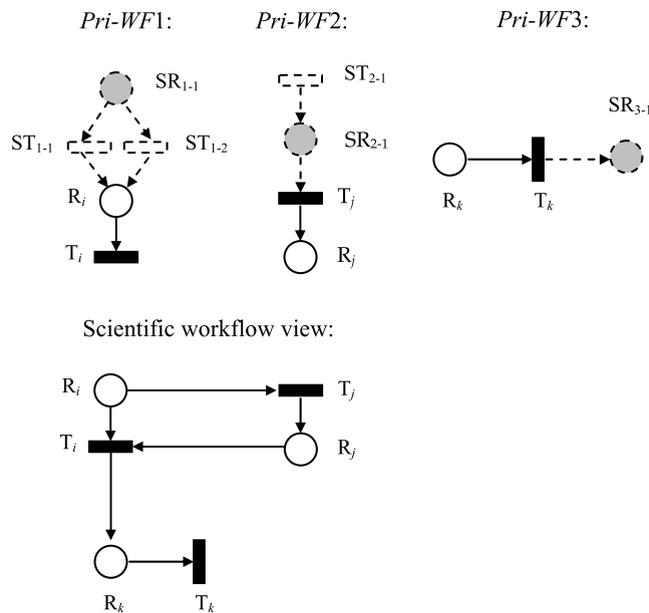


Fig. 2. Three private workflow fragments and a scientific workflow view by masking its silent context.

Fig. 1 demonstrates a scientific workflow and its application context associated with three self-managing organizations. In Fig. 1, a scientific workflow consists of three tasks (i.e., T_i , T_j , and T_k) and three resources (i.e., R_i , R_j , and R_k). T_i , T_j , and T_k are respectively associated with three private workflow fragments (i.e., *Pri-WF1*, *Pri-WF2*, and *Pri-WF3*) for achieving certain local computing goals. *Pri-WF1*, *Pri-WF2*, and *Pri-WF3* are respectively deployed inside three self-managing organizations (i.e., SM-Org-1, SM-Org-2, and SM-Org-3). Obviously, this scientific workflow is typically deployed in a cross-organizational way.

For a scientific workflow specification, it cannot cover the silent actions and silent resources contained in private fragments, as they exclusively belong to self-managing organizations for certain privacy or security reasons. Fig. 2 illustrates some private workflow views of *Pri-WF1*, *Pri-WF2*, and *Pri-WF3*, and a global scientific workflow view by masking the silent actions and silent resources engaged in *Pri-WF1*, *Pri-WF2*, and *Pri-WF3*.

In Fig. 2, each private workflow fragment is enabled by a local workflow engine. Moreover, a global workflow engine is demanded for navigating the global execution of a scientific workflow system. The local workflow engines are orchestrated by the global scientific workflow engine in a collaborative way. Their executions hold a similar architecture scenario as demonstrated in [19]. In [19], the referred scientific workflow architecture is very helpful for scientific workflow development. In the architecture, some interfaces are specified for the interoperabilities between workflow engines. In this paper, with a similar architecture scenario as presented in [19], we focus on investigating how to orchestrate the local workflow engines with a global temporal constraint specified by the global workflow engine.

From Fig. 2, we could find that *Pri-WF1*, *Pri-WF2*, and *Pri-WF3* are respectively enacted by different self-managing organizations in isolated environments. The scientific workflow only covers their publicly accessible ports. As a private workflow fragment often masks part of its own internal workflow specification and its scheduling specification, it is a challenging endeavor to coordinate the executions of a scientific workflow and a private workflow fragment at runtime for their scientific collaboration. In view of this challenge, we will firstly discuss the temporal context of a scientific workflow for its runtime scheduling.

As mentioned in [20], “scheduling deals with the assignment of jobs and activities to resources and time ranges in accordance with relevant constraints and requirements.” For a scientific workflow, its scheduling application is always promoted in a top-down way. For example, the scheduling tools such as Petri net, WF-net, or DAG [21–24] are typical associated with a direction from source behaviors to sink behaviors. They are typical a downstream scheduling style for scheduling application. In this scheduling style, the start time of the source behaviors is determined in advance, then succeeding activities are scheduled according to certain workflow patterns (e.g., And-Split, Or-Split, And-Join, and Or-Join workflow pattern [25], to name a few) and certain temporal-dependencies (e.g., Before, Meet, Overlap [26], etc.). Its scheduling application is unfolded in the same direction with its practical execution direction. Here, we take advantage of a time axis t_1 to indicate the scheduling application of a scientific workflow.

For a private workflow fragment associated with a scientific workflow, its scheduling application is different from the scientific workflow. As a private workflow fragment is always triggered by its publicly accessible port, although there are certain source behaviors and sink behaviors in its model and its later concrete execution, the concrete temporal parameters of its behaviors could not be scheduled independently. Its scheduling application is unfolded by two steps. At the first stage, according to its expected computing goal specified by a scientific collaboration, a private workflow fragment schedules its workflow model and execution in an isolated application environment. At this scheduling stage, we take no consideration of the temporal constraints of the publicly accessible ports specified by the scientific workflow scheduling specification. Here, we take advantage of a time axis t_2 to indicate the scheduling application environment of a private workflow fragment. At the second stage, taking advantage of the temporal constraints of the publicly accessible port, the temporal distributions of the private workflow fragment indicated by time axis t_2 are wholly mapped onto time axis t_1 to keep the temporal consistency with its publicly accessible port. Through time mapping, we can guarantee the temporal consistency between executions of a private workflow fragment and a scientific workflow for their scientific collaboration.

The first scheduling stage aims at specifying a private workflow fragment's internal temporal dependencies among its silent behaviors and publicly accessible ports without external temporal constraints. Its scheduling application is initiated from a certain source point. It is unfolded in the same direction with the workflow fragment's execution in practice. The second scheduling stage aims at keeping the external temporal consistency with a scientific workflow execution for certain scientific collaboration through temporal transferring from time axis t_1 to time axis t_2 . Its temporal calculating process is initiated by a publicly accessible port. It may be unfolded in a reversed direction compared with the workflow fragment's execution in practice. It is a typical hierarchical scheduling process. For example, in Fig. 1, publicly accessible ports of R_i and T_i inside *SM-Org-1* stand for a sink resource and a sink task of a private workflow fragment *Pri-WF1*. In this situation, the scheduling application of *Pri-WF1* depends on the expected start time of the scientific workflow. As the scheduling application of *Pri-WF1* is initiated by the scheduling result of the scientific workflow, their executions should be scheduled in an incorporated way.

In this section, we investigate the application context and temporal context of a scientific workflow. In the next section, we will focus on exploring a temporal reasoning rule which can coordinate the executions of a scientific workflow including private workflow fragments.

3. A temporal reasoning rule for collaborative scheduling

Firstly, suppose that there is just one publicly accessible port contained in a private workflow fragment in a self-managing organization. More complex situations would be investigated at the end of this section.

Definition 1. For a scientific workflow *SWF*, its expected executable duration could be specified by a time period of $[SWF-E_{start}, SWF-E_{end}]$, in which $SWF-E_{start}$ and $SWF-E_{end}$ respectively stand for *SWF*'s expected start time and expected end time.

Definition 2. Suppose that there is a private workflow fragment *Pri-WF_i* associated with a scientific workflow *SWF*. It has a publicly accessible port P_i engaged in *SWF*'s execution. For P_i , its expected executable duration could be indicated by a time period of $[Pri-WF_i-E_{p-start}, Pri-WF_i-E_{p-end}]$, in which $Pri-WF_i-E_{p-start}$ and $Pri-WF_i-E_{p-end}$ respectively stand for P_i 's expected start time and its expected end time.

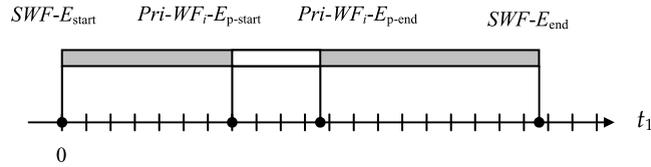


Fig. 3. Typical temporal parameters and their distributions for scheduling a scientific workflow SWF .

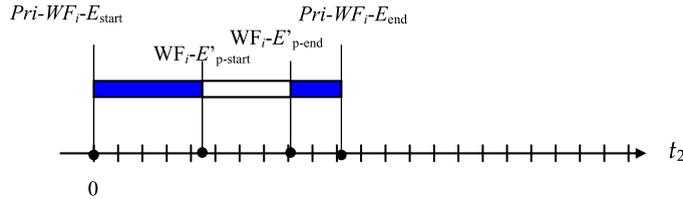


Fig. 4. Typical temporal parameters and their distributions for scheduling a private workflow fragment $Pri-WF_i$.

Here, $SWF-E_{start}$, $SWF-E_{end}$, $Pri-WF_i-E_{p-start}$, and $Pri-WF_i-E_{p-end}$ are specified by SWF 's scheduling specification. Fig. 3 indicates this unique scheduling process of SWF with a time axis t_1 . Please note that the temporal parameters indicated by time axis t_1 are relative time rather than absolute time.

In practice, $Pri-WF_i$ is uniquely associated with SWF 's execution through P_i , in which SWF plays as a service consumer and $Pri-WF_i$ plays as a service provider in their cross-domain scientific collaboration. In their service-driven scientific collaboration, as a service consumer, SWF should firstly specify its service requirement, in terms of what and when; then, as a service provider, $Pri-WF_i$ is scheduled for providing the demanded service, in time, in terms of how and when. Concretely, $Pri-WF_i$'s execution aims at providing the demanded service based on SWF 's specification in time. Once a service item is determined in terms of what, required silent resources and silent task enactments could be deployed by a self-managing organization for achieving the expected computing goal. It is associated with the first scheduling stage as mentioned in Section 2. To provide the demanded service in time, $Pri-WF_i$'s implementation should be scheduled in terms of when. It is associated with the second scheduling stage as mentioned in Section 2.

Generally, for a goal-driven workflow execution, if there is no external temporal dependency with other workflow executions, it could be scheduled based on its capacity and past experiences in a self-managing way with a special execution goal [21].

Definition 3. For a private workflow fragment $Pri-WF_i$ that takes no external temporal dependency with other workflow executions, its expected executable duration could be specified by a time period of $[Pri-WF_i-E_{start}, Pri-WF_i-E_{end}]$, in which $Pri-WF_i-E_{start}$ and $Pri-WF_i-E_{end}$ respectively stand for $Pri-WF_i$'s expected start time and its expected end time.

Fig. 4 demonstrates this unique scheduling process of $Pri-WF_i$ specified by time axis t_2 . Similarly, the temporal parameters indicated by time axis t_2 are also relative time rather than absolute time.

As $Pri-WF_i$ is uniquely associated with SWF through P_i , there are certain temporal dependencies between $Pri-WF_i$ and SWF . To provide the required computing service in time, $Pri-WF_i$ should be active in a required duration based on their temporal dependencies. $Pri-WF_i$'s start time should be deduced based on the temporal constraints of its publicly accessible port specified by SWF 's specification rather than determined independently. Here, suppose that associated with time axis t_2 , P_i 's expected start time and expected end time are respectively indicated by $Pri-WF_i-E'_{p-start}$ and $Pri-WF_i-E'_{p-end}$. Here, $Pri-WF_i-E'_{p-start}$ and $Pri-WF_i-E'_{p-end}$ should be respectively equal to $Pri-WF_i-E_{p-start}$ and $Pri-WF_i-E_{p-end}$ in terms of absolute time or in execution. To keep the temporal consistency between $Pri-WF_i$ and SWF , the time parameters $Pri-WF_i-E_{start}$, $Pri-WF_i-E_{end}$, $Pri-WF_i-E'_{p-start}$ and $Pri-WF_i-E'_{p-end}$ indicated by t_2 should be mapped to SWF 's time axis t_1 .

In view of this observation, a temporal transferring rule will be investigated in this section, for keeping temporal consistency in cross-organizational scientific collaboration.

(1) For a scientific workflow SWF , as the time period of $[SWF-E_{start}, SWF-E_{end}]$ covers the time period of $[Pri-WF_i-E_{p-start}, Pri-WF_i-E_{p-end}]$, i.e., $[Pri-WF_i-E_{p-start}, Pri-WF_i-E_{p-end}] \subseteq [SWF-E_{start}, SWF-E_{end}]$, $SWF-E_{start}$ should be determined firstly, then $Pri-WF_i-E_{p-start}$ and $Pri-WF_i-E_{p-end}$ are determined according to the values of $SWF-E_{start}$ and SWF 's internal temporal distributions. This temporal scheduling is formalized by a scheduling logic of $SWF-E_{start} \mapsto [Pri-WF_i-E_{p-start}, Pri-WF_i-E_{p-end}]$. It indicates a top-down or a global-to-local temporal reasoning path for scheduling a scientific workflow.

(2) For a private workflow fragment $Pri-WF_i$, although the time period of $[Pri-WF_i-E_{start}, Pri-WF_i-E_{end}]$ covers the time period of $[Pri-WF_i-E'_{p-start}, Pri-WF_i-E'_{p-end}]$, i.e., $[Pri-WF_i-E'_{p-start}, Pri-WF_i-E'_{p-end}] \subseteq [Pri-WF_i-E_{start}, Pri-WF_i-E_{end}]$, a different temporal scheduling logic is held. More specifically, only after the values of $Pri-WF_i-E'_{p-start}$ and $Pri-WF_i-E'_{p-end}$ are achieved based on the scheduling logic of $SWF-E_{start} \mapsto [Pri-WF_i-E_{p-start}, Pri-WF_i-E_{p-end}]$, $Pri-WF_i-E_{start}$ and $Pri-WF_i-E_{end}$ could be deduced based on the concrete values of $Pri-WF_i-E'_{p-start}$, $Pri-WF_i-E'_{p-end}$ and $Pri-WF_i$'s internal temporal distributions. Here,

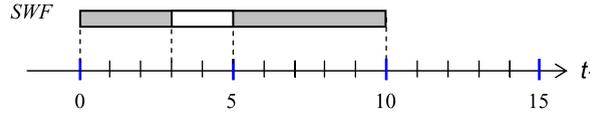


Fig. 5. Temporal parameters and their distributions of a scientific workflow example SWF associated with time axis t_1 .

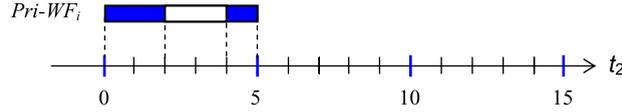


Fig. 6. Temporal parameters and their distributions of a private workflow fragment example $Pri-WF_i$ associated with time axis t_2 .

Fig. 4 illustrates $Pri-WF_i$'s internal temporal distributions in a qualitative way. This temporal scheduling process could be formalized by a scheduling logic of $[Pri-WF_i-E'_{p-start}, Pri-WF_i-E'_{p-end}] \mapsto [Pri-WF_i-E_{start}, Pri-WF_i-E_{end}]$. It indicates a bottom-up or a local-to-global temporal reasoning path for scheduling a private workflow fragment in an incorporated scheduling environment. It is different from the global-to-local temporal reasoning path.

$Pri-WF_i-E'_{p-start}$ and $Pri-WF_i-E'_{p-end}$ should be respectively equal to $Pri-WF_i-E_{p-start}$ and $Pri-WF_i-E_{p-end}$ in terms of absolute time. Accordingly, the temporal association relation between SWF and WF_i is specified by Definition 4.

Definition 4. The temporal dependency around a publicly accessible port between SWF and $Pri-WF_i$ could be formalized by $[SWF-E_{start}, SWF-E_{end}] \mapsto [Pri-WF_i-E_{p-start}, Pri-WF_i-E_{p-end}] \mapsto [Pri-WF_i-E_{start}, Pri-WF_i-E_{end}]$ for keeping temporal consistency in cross-organizational scientific collaboration.

The scheduling algorithm specified by Definition 4 could be formulated as follows.

```

To determine the expected start time and the expected end time of
a global scientific workflow  $SWF$ , i.e.,  $SWF-E_{start}$  and  $SWF-E_{end}$ .
If (a private workflow fragment  $Pri-WF_i$ , is engaged in the execu-
tion of the global scientific workflow through a public accessi-
ble port)
{To determine the expected start time and the expected end
time of  $Pri-WF_i$ 's public accessible port, i.e.,  $Pri-WF_i-E_{p-start}$  and
 $Pri-WF_i-E_{p-end}$ , according to  $SWF$ 's internal temporal distribu-
tion;
//corresponding to the operation of  $[SWF-E_{start}, SWF-$ 
// $E_{end}] \mapsto [Pri-WF_i-E_{p-start}, Pri-WF_i-E_{p-end}]$ 
To determine the  $Pri-WF_i$ 's expected start time and the ex-
pected end time, i.e.,  $Pri-WF_i-E_{p-start}$  and  $Pri-WF_i-E_{p-end}$ , accord-
ing to  $Pri-WF_i$ 's internal temporal distribution;
//corresponding to the operation of  $[Pri-WF_i-E_{p-start}, Pri-WF_i-E_{p-}$ 
// $end] \mapsto [Pri-WF_i-E_{start}, Pri-WF_i-E_{end}]$ 
}
    
```

Here, the time complexity of the scheduling algorithm would be analyzed to evaluate the feasibility of our proposal. Essentially, this scheduling algorithm specifies a linear reasoning process. Let Set-P be a set of public accessible port, and let $|Set-P| = n$. For this linear reasoning process, its time complexity is $o(n)$. Compared to the related algorithms as presented in [21–24] (e.g., Petri net), time complexity of our approach is degraded in a limited scope, which is helpful for receding state explosion problem as mentioned in [27].

To validate our algorithm, an example is presented to demonstrate the temporal transferring process specified by scheduling algorithm presented above.

Figs. 5 and 6 illustrate the temporal distributions scheduled inside an SWF 's and a $Pri-WF_i$'s internal execution. These two scheduled temporal distributions are respectively associated with time axes t_1 and t_2 . More specifically, in Fig. 5, $SWF-E_{start} = 0$, $SWF-E_{end} = 10$, $Pri-WF_i-E_{p-start} = 3$, and $Pri-WF_i-E_{p-end} = 5$; in Fig. 6, $Pri-WF_i-E_{start} = 0$, $Pri-WF_i-E_{end} = 5$, $Pri-WF_i-E'_{p-start} = 2$, and $Pri-WF_i-E'_{p-end} = 4$.

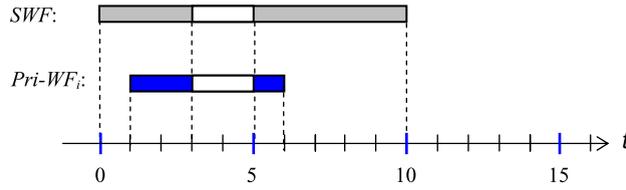


Fig. 7. Re-specified temporal parameters and their distributions of example SWF and $Pri-WF_i$ associated with a united time axis t based on their temporal dependent relation.

Here, a temporal association relation is taken into consideration between these two time axes. The incorporated temporal scheduling environment should be specified by a united time axis. For brevity and without the loss of generality, the time axis t_1 would be selected as a united time axis of t . Here, some duration parameters associated with $Pri-WF_i$ are specified, as below, for later temporal reasoning:

- (1) the duration between $Pri-WF_i-E_{start}$ and $Pri-WF_i-E'_{p-start}$ is 2 time units, i.e., $Pri-WF_{i-d1} = 2$ time units;
- (2) the duration between $Pri-WF_i-E'_{p-start}$ and $Pri-WF_i-E'_{p-end}$ is 2 time units, i.e., $Pri-WF_{i-d2} = 2$ time units; and
- (3) the duration between $Pri-WF_i-E'_{p-end}$ and $Pri-WF_i-E_{end}$ is 1 time units, i.e., $Pri-WF_{i-d3} = 1$ time units.

According to these parameters, the time parameters of $Pri-WF_i$ could be re-specified, as below, in the united time axis t . Fig. 7 illustrates the re-specified temporal parameters and their distributions in the united time axis t :

- (1) $Pri-WF_i-E'_{p-start} = Pri-WF_i-E_{p-start} = 3$;
- (2) $Pri-WF_i-E'_{p-end} = Pri-WF_i-E_{p-end} = 5$;
- (3) $Pri-WF_i-E_{start} = Pri-WF_i-E_{p-start} - Pri-WF_{i-d1} = 3 - 2 = 1$;
- (4) $Pri-WF_i-E_{end} = Pri-WF_i-E_{p-end} + Pri-WF_{i-d3} = 5 + 1 = 6$.

In Fig. 7, $Pri-WF_i$'s start time (i.e., $Pri-WF_i-E_{start} = 1$) as specified by time axis t is an ideal start time for producing the required service item for SWF 's execution. Otherwise, $Pri-WF_i$ will occupy some additional time costs or cannot provide the demanded service item in time. For example, if $Pri-WF_i$ starts at the zero time point in time axis t , i.e., $Pri-WF_i-E_{start} = 0$, as P_i expected start time is fixed in SWF 's specification, i.e., $Pri-WF_i-E_{p-start} = 3$, and $Pri-WF_i-E_{p-end} = 5$ could not be changed, the duration between $Pri-WF_i-E_{start}$ and $Pri-WF_i-E_{p-start}$ is 3 time units, i.e., $Pri-WF_{i-d1} = 3$ time units. Obviously, it wastes 1 time unit compared to T_{j-d1} 's original value (i.e., $T_{j-d1} = 2$) that we calculated previously. On the other hand, if $Pri-WF_i$ starts at the 2nd time point in time axis t , i.e., $Pri-WF_i-E_{start} = 2$, as the duration from $Pri-WF_i-E_{start}$ to $Pri-WF_i-E_{p-start}$ is a fixed value, according to their relative time distributions, $Pri-WF_i-E_{p-start}$ should be at the 4th time point, i.e., $Pri-WF_i-E_{p-start} = 4$, to meet $Pri-WF_i$'s workflow specification. Obviously, it delays the service invocation for satisfying $Pri-WF$'s execution.

This example demonstrates a real-time application for scientific collaboration. In practice, the execution of a scientific workflow system may be a mixture of hard real-time applications and soft real-time applications. Generally, a system is said to be real-time if the total correctness of an operation depends not only upon its logical correctness, but also upon the time in which it is performed [28]. Moreover, in a hard or immediate real-time system, the completion of an operation after its deadline is considered useless. On the other hand, a soft real-time system will tolerate such lateness and take the overhead of context switching into consideration. Soft real-time systems are typically useful if there are some issues of concurrent access that need to keep a number of connected systems up to date with changing situations.

To incorporate the soft real-time property into workflow scheduling application, a publicly accessible port should have some typical attributes of Has-Earliest-Start-Time, Has-Latest-Start-Time, Has-Earliest-End-Time, and Has-Latest-End-Time as specified in [20]. Moreover, a temporal-dependable service initiated by a publicly accessible port subscribes to Allen's [26] representation of standard time and relations between a private workflow fragment and a scientific workflow. These temporal attributes are key temporal constraints for task enactment and resource allocation for scientific collaboration.

Here, some more complex situations would be investigated. Suppose that there is more than one publicly accessible port contained in a self-managing organization. With this scenario in our mind, some complex situations are distinguished as below.

(1) If the ports belong to a same private workflow fragment $Pri-WF_i$, and $Pri-WF_i$ just engaged in a scientific collaboration with a scientific workflow, its local temporal scheduling among its silent actions, silent resources and publicly accessible ports just aims at providing the demanded service item in time. In this situation, there is no conflict in resource sharing and task enactment, and it is easy to schedule a $Pri-WF_i$ in a self-managing way.

(2) If the ports belong to a same private workflow fragment $Pri-WF_i$, and $Pri-WF_i$ is engaged in more than one scientific workflow in a concurrent environment, its local temporal scheduling among its silent actions, silent resources and publicly accessible ports should be coordinated with each other for satisfying different service items for different scientific workflows.

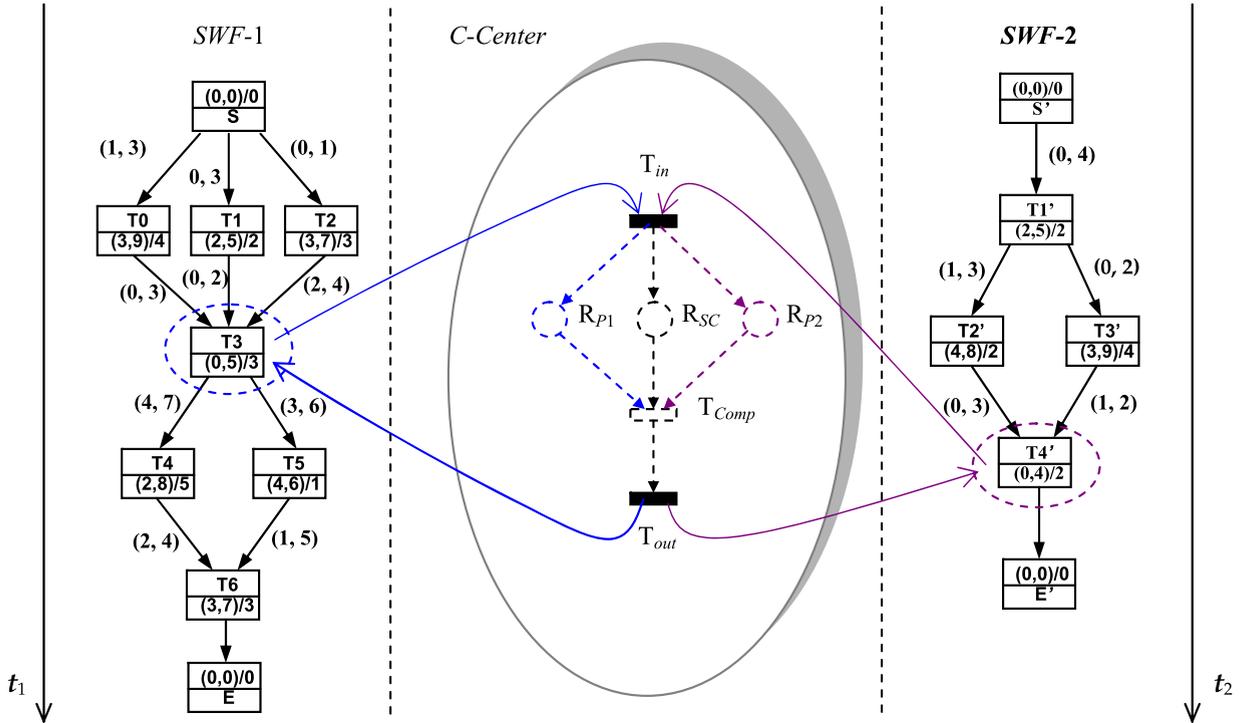


Fig. 8. Coordination of two scientific collaborations between a self-managing organization and two scientific workflows SWF-1 and SWF-2.

It is required that $Pri-WF_i$ should compromise the service producing processes if there is a conflict in resource sharing and task enactment.

(3) If the ports belong to different private workflow fragments, and there is no shared silent action or silent resource for producing the service items among the private workflow fragments, the private workflow fragments could be scheduled independently with each other, according to the temporal transferring rule proposed in this paper.

(4) If the ports belong to different private workflow fragments but there are some shared silent actions or silent resources among the workflow fragments, the scheduling application of the workflow fragments should be promoted in an incorporated way. In the following section, an evaluation would be presented to demonstrate this complex situation based on the method presented in this section.

4. Evaluation

In this section, we will illustrate our approach using a complex example. Suppose there are two individual task-driven scientific workflows SWF-1 and SWF-2. Moreover, a self-managing computing center C-Center is engaged in these two scientific workflow executions in a concurrent application environment. The executions of SWF-1, SWF-2, and C-Center own the soft real-time property. Fig. 8 illustrates the application context. To process the service requirement of SWF-1 (i.e., Serviceltem-1), C-Center will recruit a simulation and analysis program P_1 and a super-computer SC that are managed by C-Center. Similarly, to process the service requirement of SWF-2 (i.e., Serviceltem-2), C-Center will recruit a simulation and analysis program P_2 and the super-computer SC that are also managed by C-Center. Associated with these two different service items, there are two private workflow fragments **Pri-WF1** and **Pri-WF2** that are deployed inside C-Center. They share two publicly accessible ports of T_{in} and T_{out} for data input and data output. Fig. 9 illustrates these two private workflow fragments. Here, the super-computer SC could not simultaneously execute two programs, i.e., SC could be only occupied by one program at runtime. It is required that P_1 and P_2 be executed asynchronously.

Here, program P_1 , program P_2 and super-computer SC are silent resources, and their allocations are enacted by C-Center in a self-managing way. In practice, as the scheduling applications of SWF-1 and SWF-2 are promoted independently with each other, they are not aware of the potential conflicts in their concurrent running environment. For example, a conflict would occur if SWF-1 and SWF-2 initiate their scientific collaborations with C-Center at a same time period. By incorporating the scheduling specifications of SWF-1 and SWF-2 into a united application, the C-Center could forecast if there is a conflict or not in its resource invocation. If there is a conflict, it would coordinate the usage of the shared resource in a compromising way between SWF-1 and SWF-2 or do some compensation for its service delay. In this situation, SWF-1 and SWF-2 could be award in advance whether their service requirement could be satisfied in time or not in their future executions. Furthermore, if one service requirement could not be satisfied in time, SWF-1 or SWF-2 could know in

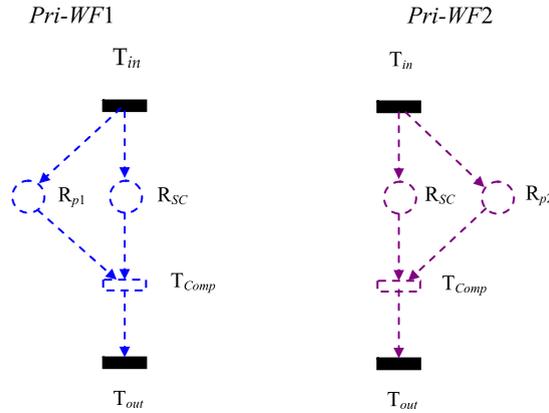


Fig. 9. Two private workflow fragments *Pri-WF1* and *Pri-WF2* respectively engaged in scientific workflows *SWF-1* and *SWF-2*.

Table 1

Calculating process for T3's active duration.

<i>SWF-1</i> : S → T3 (associated with time axis t_1)				
Path1	S → T0 → T3	T3's Has-Earliest-Start-Time $1 + 3 + 4 + 0 = 8$	T3's Has-Latest-End-Time $3 + 9 + 3 + 5 = 20$	
Path2	S → T1 → T3	T3's Has-Earliest-Start-Time $0 + 2 + 2 + 0 = 4$	T3's Has-Latest-End-Time $3 + 5 + 2 + 5 = 15$	Conclusion: T3's practical active duration should be [8, 20]
Path3	S → T2 → T3	T3's Has-Earliest-Start-Time $0 + 3 + 3 + 2 = 8$	T3's Has-Latest-End-Time $1 + 7 + 4 + 5 = 17$	

advance how long it would be delayed according to *C-Center's* coordination. It guarantees that coordinated executions could be readily available in a computation- and data-rich concurrent environment for cross-domain scientific collaboration, which greatly enhances *C-Center's* QoS-related service capability. Otherwise, an exception would occur at runtime when a conflict arises in a concurrent environment.

In Fig. 8, time axes t_1 and t_2 respectively indicate two individual scheduling applications of *SWF-1* and *SWF-2*. Here, the temporal specifications of *SWF-1* and *SWF-2* subscribe to Li's [21] temporal definition in formalization, in which the temporal specification subscribes to relative time rather than absolute time in their scheduling applications. More specifically, (2, 4) on the edge from T2 and T3 specifies the temporal dependency relation between activities T2 and T3 as an external time constraint. It means that T3 just could start between 2 time units after T2 ends and 4 time units after T2 ends. They essentially indicate T3's Has-Earliest-Start-Time and Has-Latest-Start-Time associated with activity T2. T3's (0, 5)/3 specifies its internal time constraints, i.e., its defined executable time span is $[T_1 + 0, T_1 + 5]$ if it starts enabling at time T_1 ; and its execution duration is 3. T2's internal time constraints can be interpreted similarly. If T2 completes its execution at time T_0 , the start enabled time span of T3 is $[T_0 + 2, T_0 + 4]$, and T3's enabled time span is $[T_0 + 2, T_0 + 4 + 3]$. In a run-time environment, because of the time constraint imposed by the enabled time span, T3's actual executable time span will be $[T_0 + 2, \text{Min}\{T_0 + 4 + 3, T_1 + 5\}]$, where $T_0 + 2 \leq T_1 \leq T_0 + 4$. These temporal parameters associated with activity T2 or T3 are essentially internal time constraints. With these internal time constraints, T3's Has-Earliest-End-Time and Has-Latest-End-Time properties could be deduced based on these external and internal time constraints.

Additionally, in Fig. 8, for *SWF-1*, the workflow pattern of T0, T1, and T2 is an AND-Split style initiated by S, and the workflow pattern of T0, T1, and T2 is an AND-Join style for reaching T3; for *SWF-2*, the workflow pattern of T2', and T3' is an AND-Split style initiated by T1', and the workflow pattern of T2', and T3' is an AND-Join style for reaching T4'. To forecast if there is a conflict or not in resource invocation, *C-Center's* self-managing resource allocation should be promoted through coordinating the temporal specification of *SWF-1* and *SWF-1* in an incorporated scheduling application.

From S to T3, there are three execution paths. According to the timing parameters indicated by the execution path from S to T3 through T0, we could deduce that T3's Has-Earliest-Start-Time is 8 and T3's Has-Latest-End-Time is 20 specified by time axis t_1 . It indicates that T3's expected execution should be occurred during the time period of [8, 20] associated with this execution path. Similarly, associated with the execution path from S to T3 through T1, we could deduce that T3's expected execution should be occurred during the time period of [4, 15]; associated with the execution path from S to T3 through T2, T3's expected execution should be occurred during the time period of [8, 17]. As the workflow pattern of T0, T1, and T2 is an AND-Join style for reaching T3, T3's practical active duration should be the time period of [8, 20]. It also indicates that the private workflow fragment *Pri-WF1* should be executed during this time period. Table 1 demonstrates the calculating process for T3's active duration along different paths. According to same deducing rule, Table 2 demonstrates

Table 2

Calculating process for T4's active duration.

SWF-2: S → T4' (associated with time axis t ₂)				
Path1	S' → T1' → T2' → T4'	T4's Has-Earliest-Start-Time 0 + 2 + 2 + 1 + 4 + 2 + 0 = 11	T4's Has-Latest-End-Time 4 + 5 + 3 + 8 + 3 + 8 = 31	Conclusion: T4's practical active duration should be [12, 31]
Path2	S' → T1' → T3' → T4'	T4's Has-Earliest-Start-Time 0 + 2 + 2 + 0 + 3 + 4 + 1 = 12	T4's Has-Latest-End-Time 4 + 5 + 2 + 9 + 2 + 8 = 30	

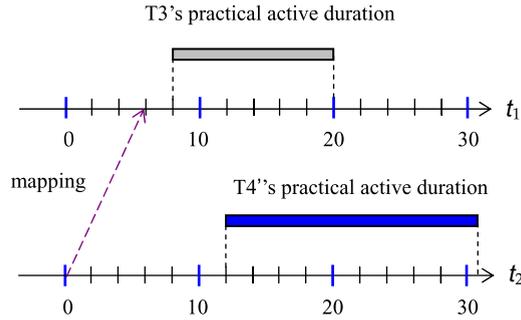


Fig. 10. T3's and T4's practical active durations associated with their individual time axes t₁ and t₂.

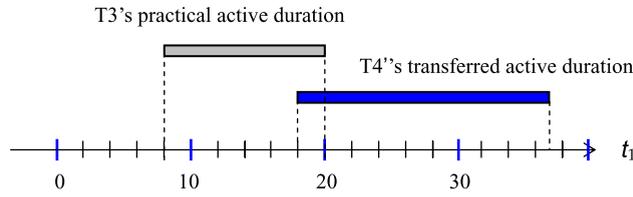


Fig. 11. Temporal transferring result from time axis t₂ to time axis t₁ with an assumption that the origin of time axis t₂ is equal to the 6th time point indicated by time axis t₁.

the calculating process for T4's active duration along different paths. Fig. 10 demonstrates T3's and T4's practical active durations respectively associated with time axes t₁ and t₂.

Fig. 10 demonstrates T3' and T4's practical active durations in their individual time axes t₁ and t₂. As these deducing processes subscribe to relative time rather than absolute time, we could not conclude that there would be a conflict around SC's invocation based on the overlapped time period. However, if we could map the time parameters indicated by t₂ onto t₁ according to the temporal transferring rule as proposed in Section 3, we could definitely determine whether there is a conflict or not around SC's invocation. Furthermore, we could determine that during which time period there may be a conflict according to the overlapped time period of the re-specified active durations. Fig. 11 demonstrates a temporal transferring result from t₂ to t₁ with an assumption that the origin of t₂ is equal to the 6th time point in t₁. With this precondition, the active duration indicated by t₂ would be transferred into the time period of [18, 37] indicated in t₁. According to the transferred temporal distribution, we could find that during the time period of [18, 20] there may be a conflict in resource sharing between SWF-1 and SWF-2.

Once the overlapped duration is determined, C-Center could coordinate the usage of the shared resource in a compromising way between SWF-1 and SWF-2 or do some compensation for a delayed service. Furthermore, according to the delayed time (i.e., 2 time units) and the relation between price and duration, C-Center could also definitely calculate the price cost in compensating for SWF-2's delayed execution. This QoS related computing process is referred to [23] for detail.

5. Related works and comparison analysis

The scheduling issue is very important for enhancing the scalability, autonomy, quality and performance of scientific workflows [4,7,10,15,20,30]. For example, in [10], three major categories of scientific workflow scheduling architecture are presented, i.e., centralized, hierarchical and decentralized scheduling schemes. In the centralized workflow enactment environment, one central scheduler makes scheduling decision for all tasks engaged in future workflow execution. For hierarchical scheduling, there is a central manager and multiple lower-level sub-workflow schedulers. This central manager is responsible for controlling workflow execution and assigning sub-workflows to the lower-level schedulers. In contrast with the centralized and hierarchical schemes, there are multiple schedulers without any central controller in decentralized scheduling. A scheduler can communicate with others and schedules a sub-workflow to other schedulers with lower load.

In [10], the authors believed that the centralized scheme can produce efficient schedules because it has all necessary information about all tasks engaged in workflow execution. However, it is not scalable with respect to the number of task and grid resource that are generally autonomous. The major advantage of using the hierarchical architecture is that different scheduling policies can be deployed in the central manager and lower-level schedulers. However, the failure of the central manager will result in entire system failure. Decentralized scheduling is more scalable but faces more challenges to generate optimal solutions for overall workflow performance. The method presented in this paper falls into the third scheme, i.e., decentralized scheduling scheme.

On the other hand, scientific research projects aim at effectively enhancing domain-across collaboration on Internet, there is a compelling need of dedicated services to support collaborative scientific workflow execution [29]. QoS related evaluation is a key issue in this kind of service-driven application paradigm. In [29], a SOA-based infrastructure for supporting scientific collaboration is presented, in which QoS is an indispensable issue for scientific collaboration. In [31], trust-based robust scheduling method is investigated to enhance the quality of scientific workflow execution. With this scenario, the temporal scheduling method presented in this paper provides an approach to guarantee quality of collaboration from temporal coordination perspective [23].

Compared with the related works [4,7,10,15,20,29–32], the main contributions of this paper are twofold.

First, as a typical application environment, grid is an efficient infrastructure for scientific workflow development and execution. In a general grid environment, scheduling of resource allocation is an important issue for cross-organizational grid service invocation based on certain privacy and security usage policies [12–15,17]. Generally, it takes less consideration of task scheduling application (i.e., private workflow fragment scheduling) enacted inside a self-managing organization for achieving a grid service. In this paper, we incorporated the resource and task into a private workflow fragment scheduling for satisfying a demanded service item. It enhances the QoS of a cross-organizational service invocation for a scientific collaboration, through keeping the temporal consistency between a scientific workflow and the private workflow fragments.

Second, for a cross-organizational scientific collaboration, privacy and security issues are key factors that should be incorporated into concrete scheduling application. In technique, brokering strategy [13,14] or view techniques [16] have been proved as efficient approaches for dealing with this problem. In this paper, the collaboration scheduling is essentially promoted based on workflow view technique, in which publicly accessible ports play as interaction view opening for scientific workflow execution. Concretely, a scientific workflow imposes certain temporal constraints on the publicly accessible ports. The silent resources and the silent tasks engaged in a private workflow fragment are scheduled based on these temporal constraints of the publicly accessible ports. It guarantees that the scheduling application of a private workflow fragment is closely navigated by a scientific workflow scheduling application. To our best knowledge, the workflow view technique is mainly employed in cross-organizational business workflow for execution supervision. In this paper, we use this technique for collaboration scheduling of a scientific workflow, which is a novel application of workflow view technique.

Please note that the temporal parameters in our scheduling application are relative time rather than absolute time. If the temporal parameters are specified in form of absolute time, the evaluation presented in Section 4 would lose its general effects in practice. It is a limitation of our method.

6. Conclusions

In this paper, a collaborative scheduling approach is presented based on a temporal transferring rule enacted between a scientific workflow and the distributed domain-specific applications deployed in several autonomous organizations. The proposed approach aims at keeping the temporal consistency of a scientific collaboration in resource sharing and task enactments. Through an evaluation, we also demonstrate the capability of our approach for promoting multiple scientific workflow executions in a concurrent environment. This collaborative scheduling approach could also be helpful with QoS-aware middleware development for cross-organizational scientific collaborations, which will be studied as a future research topic.

Acknowledgments

This paper is partly supported by the National Science Foundation of China under Grant Nos. 60721002, 60673017 and 60736015, Jiangsu Provincial NSF Project under Grants Nos. BK2007137 and BK2008017, and program for New Century Excellent Talents in University under Grant NCET-06-0440.

References

- [1] B. Ludäscher, C. Goble, Guest editors' introduction to the special section on scientific workflows, *SIGMOD Record* 34 (3) (2005) 3–4.
- [2] S. Bowers, T.M. McPhillips, B. Ludäscher, Provenance in collection-oriented scientific workflows, *Concurrency and Computation: Practice and Experience* 20 (5) (2008) 519–529.
- [3] Z. Zhao, S. Booms, A. Belloum, et al., VLE-WFBus: A scientific workflow bus for multi e-science domains, in: *Proc. 2th IEEE Int. Conf. e-Science and Grid Computing (e-Science'06)*, Amsterdam, Netherlands, December 2006.
- [4] M. Wiecezorek, R. Prodan, T. Fahringer, Scheduling of scientific workflows in the ASKALON grid environment, *SIGMOD Record* 34 (3) (2005) 56–62.
- [5] G.C. Fox, D. Gannon, Special issue: Workflow in grid systems, *Concurrency and Computation: Practice and Experience* 18 (10) (2006) 1009–1019.
- [6] T.M. McPhillips, S. Bowers, An approach for pipelining nested collections in scientific workflows, *SIGMOD Record* 34 (3) (2005).

- [7] Y. Yan, B. Chapman, Scientific workflow scheduling in computational grids – planning, reservation, and data/network-awareness, in: Proc. 8th IEEE/ACM Int'l Conf. Grid Computing, Austin, Texas, September 2007.
- [8] A. Rygg, P. Roe, O. Wong, J. Sumitomo, GPFlow: An intuitive environment for web-based scientific workflow, *Concurrency and Computation: Practice and Experience* 20 (4) (2008) 393–408.
- [9] I. Foster, C. Kesselman, S. Tuecke, The anatomy of the grid: Enabling scalable virtual organizations, *International Journal of Supercomputer Applications* 15 (3) (2001) 200–222.
- [10] J. Yu, R. Buyya, A taxonomy of scientific workflow systems for grid computing, *SIGMOD Record* 34 (3) (2005) 44–49.
- [11] J. Yu, R. Buyya, C.K. Tham, Cost-based scheduling of scientific workflow applications on utility grids, in: Proc. 1st Int'l Conf. e-Science and Grid Computing (e-Science'05), Melbourne, Australia, December 2005.
- [12] D.K.W. Chiu, S.C. Cheung, et al., Workflow view-driven cross-organizational interoperability in a web service environment, *Information Technology and Management* 5 (3–4) (2004) 221–250.
- [13] C.L. Dumitrescu, M. Wilde, I. Foster, A model for usage policy-based resource allocation in grids, in: Proc. 6th IEEE Int'l Workshop Policies for Distributed Systems and Networks, Stockholm, Sweden, June 2005.
- [14] D.T. Liu, G.M. Abdulla, et al., Data-preservation in scientific workflow middleware, in: Proc. 18th Int'l Conf. Scientific and Statistical Database Management (SSDBM'06), Vienna, Austria, July 2006.
- [15] D.M. Batista, N.L.S. da Fonseca, F.K. Miyazawa, F. Granelli, Self-adjustment of resource allocation for grid applications, *Computer Networks* 52 (9) (2008) 1762–1781.
- [16] D. Abramson, R. Buyya, J. Giddy, A computational economy for grid computing and its implementation in the Nimrod-G resource broker, *Future Generation Computer Systems* 18 (8) (2002) 1061–1074.
- [17] E. Elmroth, J. Tordsson, Grid resource brokering algorithms enabling advance reservations and resource selection based on performance predictions, *Future Generation Computer Systems* 24 (6) (2008) 585–593.
- [18] C. Li, L. Li, A distributed multiple dimensional QoS constrained resource scheduling optimization policy in computational grid, *Journal of Computer and System Sciences* 72 (4) (2006) 706–726.
- [19] C. Lin, S. Lu, et al., A reference architecture for scientific workflow management systems and the VIEW SOA solution, *IEEE Transactions on Services Computing* 2 (2) (2009) 79–92.
- [20] D. Rajpathak, E. Motta, Z. Zdrahal, R. Roy, A generic library of problem solving methods for scheduling applications, *IEEE Transactions on Knowledge and Data Engineering* 18 (6) (2006) 815–828.
- [21] J.Q. Li, Y.S. Fan, M.C. Zhou, Timing constraint workflow nets for workflow analysis, *IEEE Transactions on Systems, Man and Cybernetics (Part A)* 33 (2) (2003) 179–193.
- [22] W.M.P. van der Aalst, The application of Petri nets to workflow management, *Journal of Circuits, Systems and Computers* 8 (1) (1998) 21–66.
- [23] L.Z. Zeng, B. Benatallah, et al., QoS-aware middleware for web services composition, *IEEE Transactions on Software Engineering* 30 (5) (2004) 311–327.
- [24] Z. Guan, F. Hernandez, et al., Grid-flow: A grid-enabled scientific workflow system with a Petri-net-based interface, *Concurrency and Computation: Practice and Experience* 18 (10) (2006) 1115–1140.
- [25] W.M.P. van der Aalst, A.H.M. ter Hofstede, A.P. Barros, Workflow patterns, *Distributed and Parallel Databases* 14 (1) (2003) 5–51.
- [26] J.F. Allen, Maintaining knowledge about temporal internals, *Communications of the ACM* 26 (11) (1983) 832–834.
- [27] W. Reisig, G. Rozenberg, The State Explosion Problem, *Lecture Notes in Computer Science*, vol. 1491, 1998, pp. 429–528.
- [28] J.W.S. Liu, *Real-Time Systems*, Pearson Education Press, 2002.
- [29] S. Lu, J. Zhang, Collaborative scientific workflows, in: Proc. 7th IEEE Int'l Conference on Web Services, Los Angeles, CA, USA, July 2009.
- [30] J. Chen, Y. Yang, Temporal dependency based checkpoint selection for dynamic verification of temporal constraints in scientific workflow systems, *ACM Transactions on Software Engineering and Methodology*, in press, accepted on June 17, 2009, available at <http://www.swinflow.org/papers/TOSEM.pdf>.
- [31] M. Wang, R. Kotagiri, J. Chen, Trust-based robust scheduling and runtime adaptation of scientific workflow, *Concurrency and Computation: Practice and Experience*, in press, accepted on January 26, 2009, available at <http://www.swinflow.org/papers/TrustSciflow.pdf>.
- [32] J. Chen, Y. Yang, Activity completion duration based checkpoint selection for dynamic verification of temporal constraints in grid workflow systems, *International Journal of High Performance Computing Applications* 22 (3) (2008) 319–329.