Work Flow Scheduling Modeling in Grid Computing Using Linear Switching State Space (LS^3)

H. Tabatabaee

***Abstract*—**By using the resources in the network, we had the opportunity to solve the great scale problems. Due to this aim, Grid technology has lots of challenges. Task scheduling is one of the most important problems in this field. Different methods for solving this problem use a wide range of techniques from classical methods which are based on list scheduling to random searching which are generally based on evolutionary algorithms. Current methods apply some extra limitations or decrease some real aspects of the problem in order to solve it. In addition, they are based on evaluating efficiency in a scholar mode and cannot reach the obvious theoretical results. In this paper we propose a novel method based on System Engineering and address the theoretical analysis problem considering the special properties of a Grid environment. In this paper we prove that how we can map standard task scheduling in grid using linear switching state space (LS^3).

***Keywords*—**Grid Computing, Linear Switching System, Modeling, Work Flow Scheduling.

I. Introduction

LOTS of applications to achieve satisfying results need to work out on data in large scale using sophisticated computations. For example, Physics[1], Seismology[2], Astronomy[3]. These applications are generally modeled in the form of a workflow containing computations tasks and the relationships between them. The workflow is presented by a tasks graph. Many of tasks in the graphs are series programs which need just some seconds to run. Some of these workflows consisting millions of tasks which the general time for their calculating is thousands of hours. Special structure of these applications let the tasks to be run in parallel, meanwhile using a really fast computer (like a super computer) avoid applying this option more than high costs brings to the system. Therefore, by transferring these kinds of applications to technologies like multi processor systems or grid and then by running them in parallel. We could achieve the wanted results in a reasonable time spending less cost. One of the fundamental requirements for performing workflows efficiently on such infrastructures is to apply an appropriate scheduler. A scheduler could be defined as a process assigning tasks related to resources which should make the total time as minimum as it can. This is a NP-complete [4] problem and could not be solved by classical approaches. This problem is one of the most interesting problems for researches which leaded to be solved in different ways and by different technologies (multi processors systems, clusters, grid, etc.). Because of these efforts, today there are lots of algorithms in this field. However, many of these approaches are based on exploiting and random searching which do not have good theoretical basis. In our previous work, we addressed the theoretical analysis problem presenting a novel approach based on System Engineering in the context of multi processors systems. The novel modeling method is willing because of its theoretical extensions[5]. Considering special characteristics in a grid system, we present the same approach in the context of a grid system. A grid environment makes the task scheduling problem much harder because of its properties.

The rest of the paper is organized as follows. Section II introduces the basic concepts of Grid environments and also the challenges in scheduling tasks in Grid environments. In Section III similar works in this area are inspected. Section IV talks about the proposed model and its relationship to state model. In Section V the proposed scheduling approach is introduced in details. Finally Section VI concludes the paper and provides some further extensions on the proposed model.

II. Grid

Powerful computers and high speed networks with low costs are changing the ways computers are used. Grid could be considered as such technology which is developed for using potential powers of the resources in the network environments. Grid gives the users the opportunity to share and access to the distributed resources in a transparent and the same form. Users by using the services provided by grid middleware perform their applications in this environment. Using scheduling algorithms for performing the tasks efficiently is very important. The process of scheduling in the grid is done in 3 steps: 1) finding resources 2) selecting the resource based on the goals, and 3) sending tasks[2]. Figure 1 illustrates a model of scheduling system which shows the components are connected with two data flows: information flows of the application or the resource and the flows for the tasks or the commands of task scheduling.

Grid scheduling (GS) receives the applications from the users, selects the appropriate resources for these applications based on the gathered information from the information service module of the grid and ultimately, maps the applications to the resources based on the goals and the estimated efficiency of the resources. Unlike the current scheduling in classical parallel and distributed systems, a grid scheduling usually cannot control the resources directly, but it works like resource broker.

H. Tabatabaee is with the Department of Computer Engineering, Quchan Branch, Islamic Azad University, Quchan, Iran.

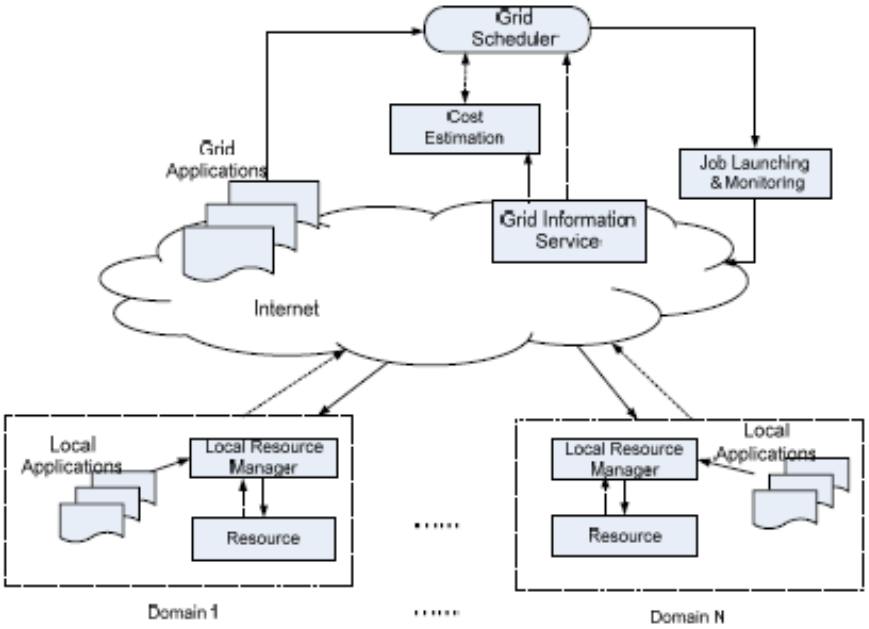


Fig. 1 A logical architecture of Grid scheduling. Dashed lines show the information flows of applications or resources. Solid lines show the task flows or the commands of task scheduling.

Grid scheduling is not essentially in the same domain with the visible resources. Figure 1 shows one grid scheduling, but indeed, several scheduling could be applied in order to construct different structures such as central, hieratical and non-central [6] base on different aims like efficiency or scalability.

For an appropriate scheduling, Grid scheduler needs to know the information about the status of the current resources, especially when the dynamic and heterogeneous nature of the grid is taken into account. The role of the grid information services is to provide such information for the scheduler. These services are responsible for gathering information and estimating the status information like CPU capacity, memory space, network band width, the software and the site load in a particular time. Grid information services can response to the queries for the resources information or they can send this information for the applicants.

A. Grid scheduling challenges

In classical parallel systems and distributed systems, Scheduling algorithms are studied wildly, like Symmetric Multiple Processors machines (SMP) and Cluster of Workstations (COW). Although these studies give us a good view to the problem, but generally produce a weak scheduling since the hypotheses considered in the classical systems are as follows.

- All resources are in one management domain.
- The scheduler could control all resources.
- The source is not changeable.
- The conflicts of input applications are managed by scheduler by following a series of policies. Therefore the impact of them on efficiency is predictable.
- The computations and their data are located in one site or data transferring is a predictable process.

Grid has some spectacular characteristics leading the design of scheduling algorithms incredibly challenging. These properties are as below ([7]):

Important primary definitions in the field of task scheduling and scheduling model are presented.

B. Primary Definitions

In scheduling of the tasks which are in the form of work flow, the goal problem which should be scheduled is presented by a task graph.

Task Graph: a task graph G is a Directed Acyclic Graph (DAG) $G = (V, E, \lambda, c)$ which shows program P like a graph model. The nodes in V show tasks in P and edges in E show the relationships between tasks. Edge $e_{ij} \epsilon E$ from n_i to n_j which $n_i, n_j \in V$, shows the information volume from n_i to n_j. The positive weight $\lambda(n)$ for the node $n \in V$ shows the number of instructions that the processor should perform to complete the task. And the non-negative weight $c\ (e_{ij}{}^k)$ for the edge $e_{ij} \epsilon E$ shows the cost of communication in the k^{th} step which could vary in the non-static environment during the time (dynamic).

A task is a set of instructions which should be performed sequentially without preemption. Before performing a node (task), all predecessor nodes should be completed and all of the inputs should be received. After performing a task, the output is ready to be transferred to the successor nodes immediately[8]. The set of all predecessor and direct nodes of n_i, $\{n_x \in V: e_{xi} \in E\}$, is showed by $pred(n_i)$, and the set of all immediate successor nodes of n_i, $\{n_x \in V: e_{ix} \in E\}$, is presented as $succ(n_i)$.

If we present the start time and the final time of the task ni as ST(ni) and FT(ni), respectively, then the scheduling length is definable as maxi{FT(ni)}. The goal of scheduling the DAG is to minimize the length of scheduling without any binding to the dependency limitations between tasks. Notice that the weight of nodes and edges are usually approximate which are achieved as soon as determining the environment specifications (CUP, memory, delay, band width, etc.).

In this paper, the destination computation environment is a repository of heterogeneous resources. The resources could be addressed as Target Processing Entity (TPE) which is a computational unit. A TPE could be a personal computer, a processor or a cluster. In the destination computation environment the resources are shared as space sharing. The scheduling of space sharing of each task is assigned to a particular set of resources during its performing period of time. After scheduling, the task does not share its assigned resources with the other tasks and cannot be preempted during performing time.

III. RELATED WORKS

Workflows scheduling is a process which manages mapping and performing related tasks on distributed systems. The scheduler assigned the proper resources for the workflow tasks so that complete the performing the workflow with minimum makespan. Generally, the problem of mapping tasks on distributed systems is NP-hard. List based algorithms such as [9], [10] and [11], some clustering based algorithms are [12], and [13], and replication-based algorithms like [14] and [15] are samples of exploiting approches for this problem. On the other hand, lots of researchers consider the problem as an optimization problem and apply random searching approaches in order to find the optimum solution. Genetic algorithm, [16], Ant colony, [17], Simulated Anealing, [18] are a few examples of the works in this area. The main shortcoming of the related works is not having the appropriate theoritical basis for analatical review of peformance.

In our previous work [5], we addressed the theoretical analysis problem presenting a novel approach based on System Engineering in the context of multi processors systems. In this paper we aim to apply this approach to the problem of scheduling the related tasks in grid in order to overcome the drawbacks of the previous works.

In contrast with the previous works that usually introduce a specific algorithm, in this paper we propose a general mathematical model based on analytical proof using state space theory that is capable of modeling a wide range of scheduling problems. Here, we consider an analytical comparison between our model and two other similar works that use control theory in their model. In 2002, Lu, et.al. [19] proposed a similar model using a feedback control loop which is invoked at every sampling instant k. The submitted tasks are then scheduled by a basic scheduler with a scheduling policy such as Earliest Deadline First (EDF). The scheduler monitors the current errors using two control variables and comparing the performance references with corresponding controlled variables. Since this work is mainly developed for real-time systems, another component of this model is QoS actuator which changes the total estimated requested utilization dynamically at each sampling instant k according to the control input variables.

Also in 2002, Marbini and Sacks [20] proposed a simple PID controller which controls CPU utilization parameter of a single-processor system for simple tasks being submitted in a uniformly distributed scheme. The tasks are assumed to be independent. The miss ratio of the accepted tasks was periodically monitored and sent to PID. The control equations associated to the PID calculates the control signal. Another component called Admission Controller (AC) exploits these results for deciding whether to accept new tasks or not.

By the time the above mentioned works use control theories for solving task scheduling problem, the scope covered by these works remained rather limited and simplified versions of scheduling, i.e. addressing only QoS in [21] and only one processing unit in [7].

IV. THE PROPOSED SCHEDULING MODEL IN STATE SPACE

We develop the state space model for task scheduling in grid as below:

$$X[k+1] = max\left(0, X[k] - \Delta T * Diag(\left[\frac{1}{\theta_{ij}}\right] * U)\right) \quad (1)$$

In the above formula, ΔT is the time step in ms and Diag is the primary diameter of matrix A. Other symbols in formula 1 are presented further [22]:

A. Status variables

Status variables in this model are corresponding to the vector X which its scale is equal to the number of tasks and each item has a value between 0 which means a completed task and 1 which represents the task is not started. Therefore, the element $x_j[k]$ corresponds the remained process of a given task j in the k^{th} step.

$$X[k] = [x_1[k] \quad x_2[k] \quad . \quad . \quad . \quad x_T[k]]' \quad (2)$$

In which T=|V| and all elements $X[k]$ are assigned to 1.

B. Control vector

Our initial proposed model could not support the changes in existence of the resources in grid. Then in order to support of this specification, we change the definition of control matrix. Matrix U[k] is applied to present the status of performing each task and each resource in k^{th} step. Because the resources could be added or removed from the grid, the current resources could be changed in different time steps. Therefore we consider control vector U[k] as a $M_k \times T$ matrix which T=|V| is the number of tasks and M_k=|R_k| is the number of resources in k^{th} time step. Each item in this matrix is between 0 and 1 and shows the processing power of each resource. The characteristic of this matrix is that in each step there is maximally a non-zero item in every row and every column. Item $u_{i,j}[k]$ shows the performance rate of resource i performing task j. if $u_{i,j}[k]=1$, it said that task j in k^{th} time step is completely performed on resource i. An important limitation in control vector is that all tasks –which are performed in k^{th} step in parallel- should be independent of each other. In other words, each task should be performed only on a resource and each resource could perform at maximally one task in every time.

C. θ matrix

Matrix $\theta = [\theta_{ij}]_{T*M_k}$ is a matrix in which the item θ_{ij} determines how long it takes, if the task number i wants to be performed on resources number j. in grid this information is obtained by using performance prediction, which is provided for resources,

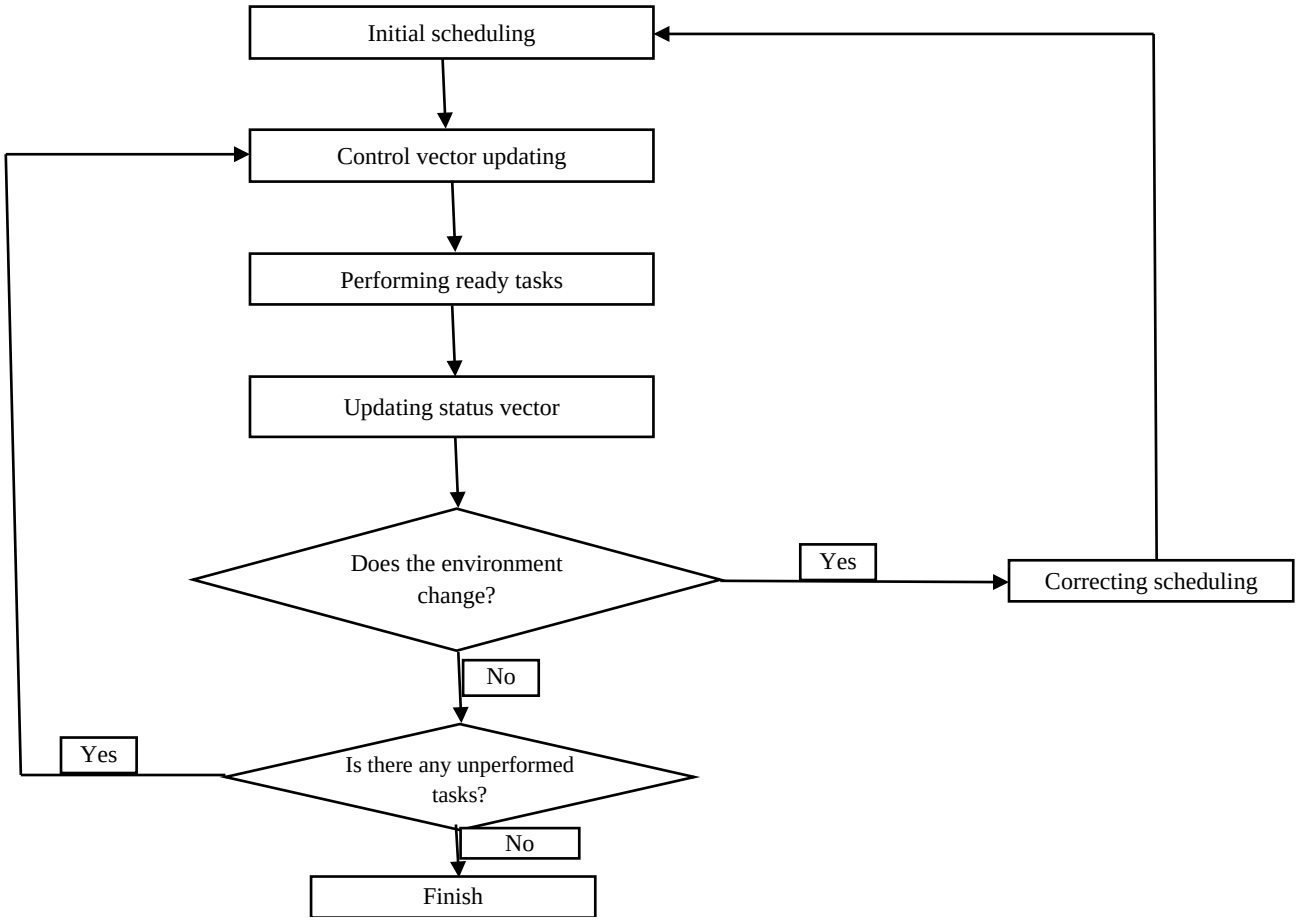


Fig. 2 scheduling steps

D. Stating the system theories for the model

Stability analysis and design of the control algorithm should incorporate the problem constraints. These constraints are summarized as follows:

1. {Initialization} The initial state vector is set to a column of ones, i.e. all of the tasks need to be executed at the beginning.
2. All of the elements of the control vector *U* are chosen in range [0,1].
3. {Constraint of Independence} In each column and each row of control vector, U, there is at most one non-zero element. In other words, there is at most one processor allocated to each task at any given time. Also, there is at most one task assigned to a processor at any given time.
4. {Constraint of Predecessors} Based on precedence relations in DAG, each task can be executed only after execution of its predecessor(s).
5. {Concurrent Task Independence} Based on precedence relations in DAG, all tasks that are executed concurrently are independent from each other.
6. {Constraint of Communication Cost} If two tasks with precedence relation are scheduled on different processors, non-zero communication cost is considered. Otherwise, communication cost is $\mu_{i,j} = 0$.
7. {Constraint of Task Completion} After completion of task execution, corresponding processor is freed.
8. {Constraint of Completeness} All of the given tasks must be assigned (and completed since a task does not free a processor until it is completely executed).

V. SCHEDULING APPROACH

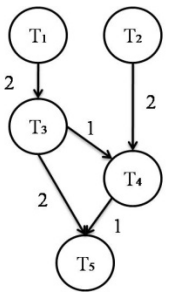
Considering grid dynamic environment, using static scheduling causes a reduction in performance; therefore, the adaptive scheduling is applied which is done in 2 phase: static scheduling and dynamic scheduling. In the first phase, an algorithm is used to find the scheduling pattern with the minimum makespan. There is no limitation in exploited algorithms; therefore, evolutionary algorithms (like HEFT) or random search algorithms (like GA) could be applied. In the second phase, the tasks are executed and the initial scheduling pattern would be corrected by scheduler, if necessary, if any similar changes occur in the environment. These phases are illustrated in figure 2. The main advantage of scheduling in 2 phases is our confidence in the initial scheduling pattern to be appropriate and in phase 2, we do not face time delay for assigning the tasks to the resources.

E. Supporting the specifications of the grid environment.

Dynamicity is the most important property of grid which differentiates it from other technologies. Therefore considering it is very essential. The first form of dynamicity is related to existence of the resources. This kind of dynamicity is because the resources in grid can exit from the grid or new resources could come. As we mentioned in section II-V, the proposed model can support the variation in existence of the resources. When a resource executing a task exits, in addition to its impact on the control vector, the status vector would be changed in the way that it reflects

stopping execution of the task. For this, the item corresponding to the task in the vector is set to 0. The second form of dynamicity in grid environment is related to performance amount which the resources present, when they are in the grid. In fact, the power amount that a resource presents during its presence in grid environment is not static and it may vary. This issue, in the worst mode, makes the task performing longer. In fact, the task, for finishing needs the resources longer than the expected time –more than the expected time steps. Therefore the control vector values – which show the map between the tasks and the resources- for this couple of the resource and the task, would not change in iterative steps –until the task performing complete.

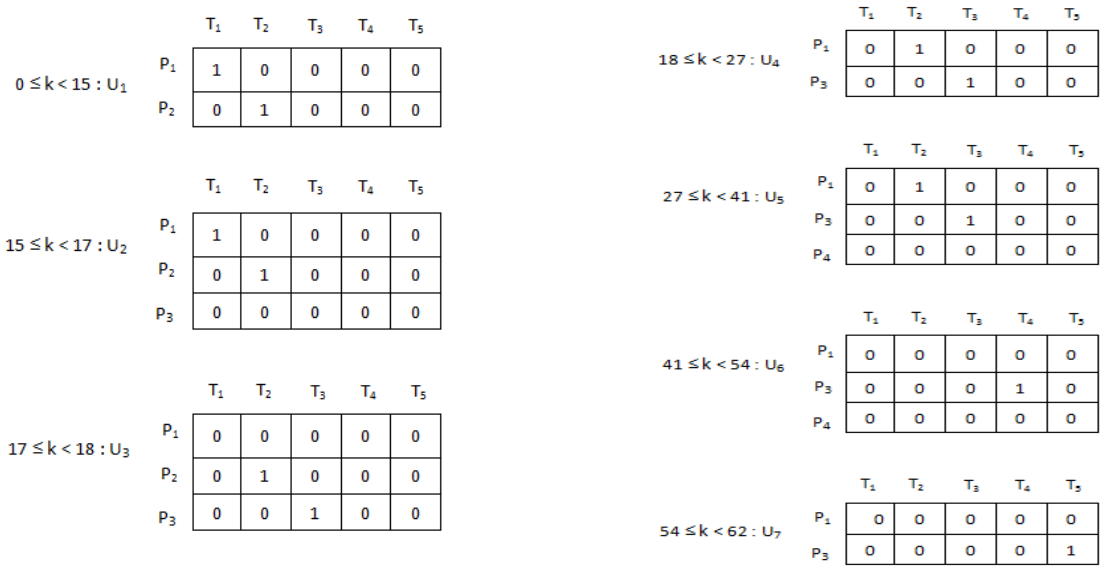
More than dynamicity of the grid environment, the proposed model can support heterogonous resources which is another characteristic of a grid. The difference between the powers that the resources have for performing tasks is showed by applying θ matrix and is reflected in the model.



(a)

θ Matrix				
	P_1	P_2	P_3	P_4
T_1	15	18	14	21
T_2	21	25	17	29
T_3	14	17	10	22
T_4	16	17	13	18
T_5	11	12	8	13

(b)



(c)

Fig. 3 a scalar example, a) DAG, b) timing table, c) control vectors

In this section we show the phases of scheduling and the reflection on the proposed model by an example. The task graph is presented in the figure 3-a. we assume that the number of resources is 4. These resources are added to the grid or exit from it according to figure 4. In this figure, the dash parts show the loss of the resources in the grid environment. For instance, the resource number 3 (P3) is not present in the environment until time 15 (d1), however it is added after that and is present by the end of performing tasks graph. Θmatrix shows task performing time on this resource.

As it can be seen in figure 3-c, considering the present resources in different time spans, control vectors change. Table I shows the status vector values in different time spans. As it was mentioned before, if a task fails or a resource performing a task exits, the task is reset to the initial status and then it would be scheduled again. For example, the second task (T2) is failed during performing because the second resource (P2) exists. And then because there is some free resources the task is scheduled immediately. Performing of this task on the new resource is starting again from the first point. Therefore the corresponding item in the status vector is set to 1.

VI. CONCLUSION AND FUTURE WORKS

We addressed the theoretical analysis of the problem of scheduling of related tasks in the context of a grid environment. In the proposed approach the problem of task scheduling is mapped to a state switching problem in control theory. In fact, by applying the proposed approach we can model dependent tasks performing in grid, not just introducing an algorithm. Also consistency of the final model is also presented using theoretical analysis. And finally it is shown that the proposed model is able to cover the specific grid characteristics for the task scheduling problem. As a future extension, we aim to present a systematic solution in order to analyze the consistency of dependent task scheduling algorithms in grid environment.

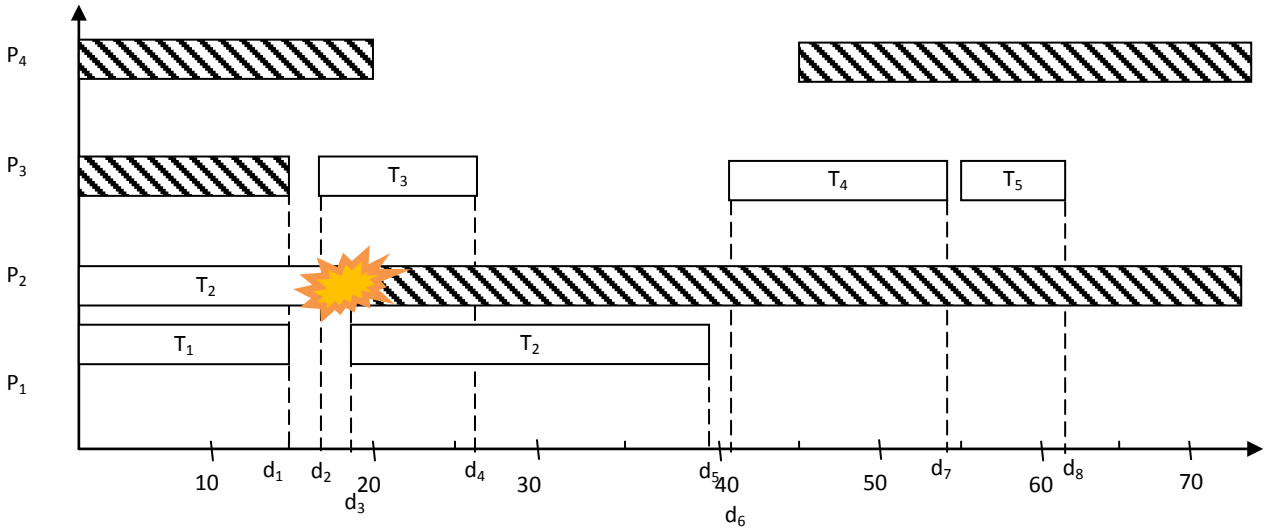


Fig. 4 the timing shape of the existence of the resources

Table I

the value of status variable

	d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8
X_1	1	0	0	0	0	0	0	0	0
X_2	1	0.4	0.32	1	0.57	0	0	0	0
X_3	1	1	1	0.9	0	0	0	0	0
X_4	1	1	1	1	1	1	1	0	0
X_5	1	1	1	1	1	1	1	1	0

REFERENCES

[1] E. Deelman, C. Kesselman, G. Mehta, L. Meshkat, L. Pearlman, K. Blackburn, P. Ehrens, A. Lazzarini, R. Williams, and S. Koranda, "GriPhyN and LIGO, building a virtual data Grid for gravitational wave scientists," in *11th IEEE International Symposium on High Performance Distributed Computing* 2002, pp. 225-234.

[2] E. Deelman, S. Callaghan, E. Field, H. Francoeur, R. Graves, N. Gupta, V. Gupta, T. H. Jordan, C.Kesselman, P. Maechling, J. Mehringer, G. Mehta, D. Okaya, K. Vahi, and L. Zhao, "Managing Large-Scale Workflow Execution from Resource Provisioning to Provenance Tracking: The CyberShake Example," in *2nd IEEE International Conference on e-Science and Grid Computing*, 2006.

[3] D. S. Katz, J. C. Jacob, E. Deelman, C. Kesselman, S. Gurmeet, S. Mei-Hui, G. B. Berriman, J. Good, A. C. Laity, and T. A. Prince, "A comparison of two methods for building astronomical image mosaics on a grid," in *International Conference on Parallel Processing*, 2005, pp. 85-94.

[4] Y. K. Kwok and I. Ahmad, "Dynamic Critical-Path Scheduling: An Effective Technique for Allocating Task Graphs to Multiprocessors," *IEEE Trans. Parallel and Distributed Systems*, vol. 7, pp. 506-521, 1996.

[5] H. Tabatabaee and M.-R.Akbarzadeh-T., "The Linear Switching State Space: A New Modeling Paradigm for Task Scheduling Problems," *Accepted in International Journal of Innovative Computing, Information and Control (IJICIC)*, 2012.

[6] V. Hamscher and U. Schwiegelshohn, "Evaluation of Job-Scheduling Strategies for Grid Computing," in *Proceedings of The 1st IEEE/ACM International Workshop on Grid Computing* 2000, pp. 191-202.

[7] Z. Sun, *Switched Linear Systems: Control and Design (Communications and Control Engineering) [Hardcover]*: Springer, 2005.

[8] O. Sinnen and e. al, "Toward a realistic task scheduling model," *IEEE Trans. Parallel and Distributed Systems*, vol. 17, pp. 263-275, 2006.

[9] H. Topcuoglu, S. Hariri, and M. Wu, "Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing," *IEEE Trans. Parallel and Distributed Systems*, vol. 3, pp. 260-274, 2002.

[10] R. Sakellariou and H. Zhao, "A Hybrid Heuristic for DAG Scheduling on Heterogeneous Systems," in *Proceedings of 18th IPDPS*, Santa Fe, New Mexico, USA, 2004, pp. 111-123.

[11] A. Radulescu and A. Gemund, "On the Complexity of List Scheduling Algorithms for Distributed Memory Systems," in *Proceedings of the 13th ISC.*, Rhodes, Greece, 1999, pp. 68-75.

[12] A. Geras, "A Comparison of Clustering Heuristics for Scheduling Directed Acyclic Graphs on Multiprocessors," *Journal Of Parallel and Distributed Computing*, vol. 4, pp. 276-291, 1992.

[13] J. Liou and M. Palis, "A Comparison of General Approaches to Multiprocessor Scheduling," in *Proceedings of the 11th International Symposium on Parallel Processing*, pp. 152-156.

[14] R. Bajaj and D. P. Agrawal, "Improving Scheduling of Tasks in A Heterogeneous Environment," *IEEE Transaction on Parallel and Distributed Systems*, vol. 2, pp. 107-118, 2004.

[15] S. Ranaweera and D. P. Agrawal, "A Task Duplication Based Scheduling Algorithm for Heterogeneous Systems," in *Proceedings of the 14th IPDPS*, Cancun, Mexico, 2000, pp. 445-450.

[16] S. Song, Y. Kwok, and K. Hwang, "Security-Driven Heuristics and A Fast Genetic Algorithm for Trusted Grid Job Scheduling," in *Proceedings of the 14th IPDPS*, Denver, Colorado USA, 2005, pp. 65-74.

[17] C. Wei-neng, S. Yuan, and Z. Jun, "An ant colony optimization algorithm for the time-varying workflow scheduling problem in grids," in *IEEE Congress on Evolutionary Computation*, 2009, pp. 875-880.

[18] S. Benedict and V. Vasudevan, "Scheduling of Scientific Workflows using Evolutionary and Threshold Accepting Algorithm for Grids," *Asian Journal of Information Technology*, vol. 8, pp. 859-865, 2007.

[19] C. Lu, J. A. Stankovic, G. Tao, and S. H. Son, "Feedback Control Real-Time Scheduling: Framework, Modeling, and Algorithms," *Special issue of Real-Time Systems Journal on Control-Theoretic Approaches to Real-Time Computing*, vol. 23, pp. 85-126, 2002.

[20] A. D. Marbini and L. Sacks, "Considering Control Theory in Resource Management Scenarios," in *Proceedings of London Communications Symposium*, London, England, 2002.

[21] O. H. Roux and A.-M. Déplanche, "A T-time Petri net extension for real-time task scheduling modeling," *"European Journal of Automation*, vol. 36, pp. 973--987, 2002.

[22] H. Tabatabaee, M.-R.Akbarzadeh-T., and N. Pariz, "Linear Switching State Space (LS3) for Fuzzy Dynamic Task Scheduling in Unstructured Heterogeneous Multiprocessor Systems," *Submitted to IEEE Transactions on Automatic Control*, 2012.