Joint Scheduling of Tasks and Communication in WDM Optical Networks for Supporting Grid Computing

Xubin Luo and Bin Wang

Department of Computer Science and Engineering
Wright State University
Dayton, OH 45435
{luo.4,bin.wang}@wright.edu

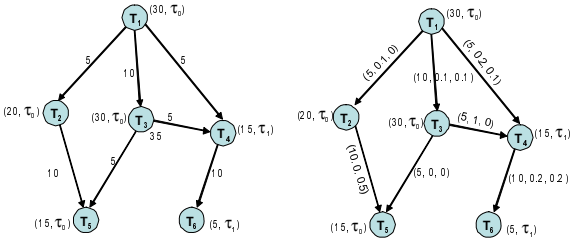
Abstract. A *flexible task model* (FTM) is proposed for modeling the relationship between grid tasks. We investigate the problem of scheduling grid computing tasks under FTM using light-trails in WDM networks to supporting the data communication between the tasks. Simulation results show that our proposed task scheduling algorithm under FTM significantly reduces the total task completion time.

1 Introduction

A grid application may consist of multiple interrelated tasks. A task model is used to capture the tasks and their interrelationships, and is commonly represented by a directed acyclic graph (DAG), also referred to as a task graph. An example DAG is shown in Figure 1(a). Conventional task models (CTM) used in most existing work assume that both task execution and data communication between tasks are atomic, in the sense that a task cannot produce its output until it has completed its execution, and that upon completion of execution, all the output is ready at that moment, and a task cannot start to execute until it receives all the inputs from its predecessors. This is reasonable because CTM was originally proposed for modeling and scheduling of the comparatively light-weighted processes of a parallel program in a multi-processor system. However, in a grid application environment, the grid tasks usually take much longer to execute and the amount of data exchanged between the tasks is much larger. Many practical scenarios exist in which a task generates output in the middle of execution or as soon as execution starts, and a task may start to execute when it receives adequate amount of (not necessarily all) inputs from its predecessors.

Therefore, a *flexible task model* (FTM) is proposed for modeling the relationship between the grid tasks. In FTM, a task may generate output before the task completes and may start to execute when it has collected a minimum amount of required input from its predecessors. For a task, the required percentage of input before starting execution and percentage of execution before generating output can be a value from 0 to 100%. Thus, FTM is more general and flexible than the conventional task graph model considered in previous work. At the

P. Primet et al. (Eds.): GridNets 2008, LNICST 2, pp. 223–230, 2009.



(a) CTM; (b) FTM.

Fig. 1. Task graph examples

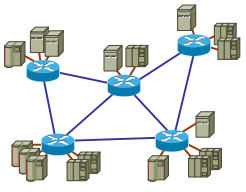


Fig. 2. A grid with five switching nodes and three types of resources

same time, we assume that the output of a task in FTM is produced as a steady stream. Figure 1(b) is an example FTM task graph. Specifically, the number 5 associated link (T_1, T_4) is the total amount of data produced by T_1 and needs to be transmitted to T_4, 0.2 is the portion of execution that T_1 has to finish before it starts to generate output for T_4, and 0.1 is the portion of data that T_4 needs to collect from T_1 before it starts to execute.

We assume that the grid is composed of two parts: the WDM optical network for provisioning communication services between the grid tasks, and the resources attached to the switching nodes in the WDM network for executing the grid tasks. Figure 2 shows an example of a grid with five switching nodes and three types of resources. We assume no communication contention in the access links, i.e., the access links connecting the resources to the switching nodes are assumed to have abundant bandwidth. Therefore, there is no communication contention within a cluster of resources connected to the same switching node.

We adopt a centralized architecture for task scheduling in which a central server is responsible for allocating the tasks to proper resources and managing the light-trails in the network. The server also keeps the status of all the resources, optical links, and the established light-trails in the optical network.

The problem considered in this paper is to determine an optimal joint schedule for executing the grid tasks and for supporting communication between tasks using light-trails such that the total time taken to complete all the tasks is minimized. The problem is $\mathcal{NP}$-complete because the problem of conventional task graph scheduling is a special case and it has been proved to be $\mathcal{NP}$-complete [1].

In [2] the communication service between the tasks is provided using lightpaths. However, a new connection requires setting up a new light-path unless there happens to be an existing lightpath with sufficient spare capacity between

the same pair of source and destination nodes. In this work, we propose to provision the communication services using light-trails [3,4,5,6] in WDM optical networks. A light-trail can support a connection even if the connection's source and destination node are in the middle of the light-trail, as long as the source node is in the upstream of the destination node and the space capacity of the light-trail is enough to support this connection. We develop scheduling algorithms to solve the joint scheduling problem. Extensive simulations are conducted and analyzed. Our algorithms show excellent performance by significantly reducing the total task completion time, compared with the approaches taken under the conventional task graph model.

The rest of this paper is organized as follows. Section 2 presents a formal definition of the problem. Section 3 analyzes the problem by interpreting the relationship between the tasks in the task graph, based on which Section 4 proposes the joint scheduling algorithm. The performance evaluation are reported in Section 5. Section 6 concludes the paper.

2 Problem Definition

In this section we formally define the scheduling problem considered in this paper. The notations used are as follows:

- $\mathcal{T} = \{T_1, T_2, \cdots, T_n\}$: the set of nodes (tasks) in the task graph;
- $\mathcal{L}$: the set of directed links in the task graph: $\mathcal{L} = \{(T_i, T_j) | T_i$ *is a predecessor of* $T_j\}$;
- $w(T_i)$: the relative execution time of task T_i;
- $t(T_i)$: the type of resource required by task T_i for execution;
- $w_{(i,j)}$: the weight of the directed link from node T_i to T_j which depicts the amount of data to be transmitted from task T_i to task T_j;
- $\alpha_{(i,j)}$: a number between 0 and 1 that specifies the percentage of the execution that task T_i needs to complete before it starts to produce output for task T_j;
- $\beta_{(i,j)}$: a number between 0 and 1 that specifies the percentage of the output from task T_i that task T_j has to collect before it starts to execute.

The WDM optical network is modeled as a graph $(\mathcal{N}, \mathcal{E})$ where $\mathcal{N}$ is the set of switching nodes and $\mathcal{E}$ is the set of optical links between the switching nodes. Each optical link e is assumed to have W wavelengths, and each wavelength has a capacity of C. Moreover, we assume no wavelength conversion. At the same time, a set of resources, $\{M_1, M_2, \cdots, \}$ are attached to the switching nodes. $t(M_h)$ indicates the resource type of resource M_h. Resource M_h is characterized by its processing power $p(M_h)$. For a task T_i compatible with M_h, the time it takes M_h to execute T_i is $w(T_i)/p(M_h)$. Furthermore, a set of light-trails $\mathcal{LT} = \{lt_1, lt_2, \cdots, \}$ will be created in the WDM network to support the communication between tasks.

The problem considered in this work is to determine a joint schedule such that the tasks are assigned to resources for execution, and the communication

channels in the WDM network are allocated for supporting the data communication between the tasks. The objective is to minimize the total amount of time to complete all the grid application tasks.

3 Problem Analysis

3.1 A Basic Case for Link (T_i, T_j)

We first examine a basic case in which (1) T_i and T_j are connected by a directed link (T_i, T_j), and (2) no other directed link leads to T_j. Suppose T_i and T_j are assigned to resource M_i and M_j, respectively, where M_i and M_j may (or may not) be the same resource.

The following notations and parameters are introduced:

- $t_s(T_i)$: the start time of task T_i;
- $t_e(T_i)$: the end time task T_i;
- $t_{os}^{(i,j)}$: the time when T_i starts to produce output for T_j. We have:

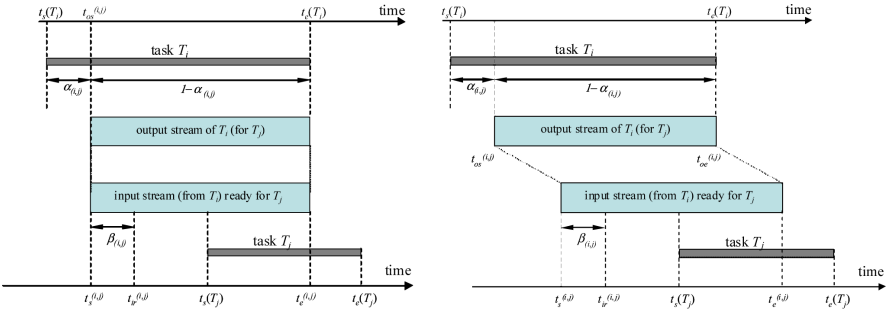
$$t_{os}^{(i,j)} = t_s(T_i) + \alpha_{(i,j)} \cdot (t_e(T_i) - t_s(T_i)) \tag{1}$$

- $r_o^{(i,j)}$: the data rate of T_i's output for T_j: $r_o^{(i,j)} = w_{(i,j)}/(t_e(T_i) - t_{os}^{(i,j)})$;
- $t_s^{(i,j)}$: the transmission start time of the data stream from task T_i to task T_j;
- $t_e^{(i,j)}$: the transmission end time of the data stream from task T_i to task T_j;
- $r^{(i,j)}$: the actual transmission data rate of the data stream from task T_i to task T_j which depends on the provisioning capability of the underlying network as well as the produced data rate of T_i. Therefore $r^{(i,j)} = \min\{r_o^{(i,j)}, C\}$;
- $t_{ir}^{(i,j)}$: the time when the data stream from task T_i to task T_j has accumulated enough to start the execution of task T_j. We have:

$$t_{ir}^{(i,j)} = t_s^{(i,j)} + \beta_{(i,j)} \cdot (t_e^{(i,j)} - t_s^{(i,j)}). \tag{2}$$

Three cases needs to be considered for the scheduling the execution of T_j as well as the data transmission between T_i and T_j:

- **Case 1:** $M_i = M_j$. Obviously we have $t_s(T_j) \geq t_e(T_i)$.
- **Case 2:** $M_i \neq M_j$ and they are located in the same cluster. The execution sequence of the two tasks is shown in Figure 3(a). The execution of tasks and output/input data streams are represented by rectangles, whose lengths correspond to the time durations. In this case the only requirement is $t_s(T_j) \geq t_{ir}^{(i,j)}$ where $t_{ir}^{(i,j)}$ can be obtained by Eq. (2), in which $t_s^{(i,j)} = t_{os}^{(i,j)}$ and $t_e^{(i,j)} = t_e(T_i)$.
- **Case 3:** $M_i \neq M_j$ and they are located in different clusters. The execution sequence of the two tasks is shown in Figure 3(b). In this case a *light-trail* is needed in the WDM network to accommodate the data stream from T_i to T_j as soon as possible and no earlier than $t_{os}^{(i,j)}$. The light-trail can be an



(a) Case 2 (b) Case 3

Fig. 3. The execution sequence of tasks T_i and T_j

existing one or a newly established one. Its spare capacity should be no less than $r^{(i,j)}$ in a time interval of length $w_{(i,j)}/r^{(i,j)}$. Because $t_s^{(i,j)}$ is the time at which a light-trail is available to accommodate the data stream from T_i to T_j, we have $t_s^{(i,j)} \geq t_{os}^{(i,j)}$ and $t_e^{(i,j)} = t_s^{(i,j)} + w_{(i,j)}/r^{(i,j)}$. More details are explained in Section 4.

When the incoming data are ready ($t_{ir}^{(i,j)}$), we need to check the availability of M_j. Therefore, we maintain the availability of each resource over time. Based on the availability information, we can then determine a value for $t_s(T_j)$ such that $t_s(T_j) \geq t_{ir}^{(i,j)}$ and during $[t_s(T_j), t_s(T_j) + w(T_j)/p(M_j)]$ the resource is available.

$t_e(T_j)$ depends on the actual execution time of T_j as well as the input data stream, whichever ends later. As a result, we have

$$t_e(T_j) = \max\{(t_s(T_j) + w(T_j)/p(M_j)), t_e^{(i,j)}\}. \tag{3}$$

3.2 Task Scheduling with Multiple Predecessors

For a task T_j with multiple predecessors, the start time $t_s(T_j)$ is constrained by the data streams from all of its predecessors, and only when all of the constraints are met can T_j start to execute. Therefore, we have Eqs (4) and (5) for $t_s(T_j)$:

$$t_s(T_j) \geq \max_{\forall i,(T_i,T_j)\in\mathcal{E},M_i\neq M_j} t_{ir}^{(i,j)}. \tag{4}$$

$$t_s(T_j) \geq t_e(T_i), \text{ if } M_i = M_j. \tag{5}$$

and Eq (6) for $t_e(T_j)$:

$$t_e(T_j) = \max\{(t_s(T_j) + w(T_j)/p(M_j)), \max_{\forall i,(T_i,T_j)\in\mathcal{L}} t_e^{(i,j)}\}. \tag{6}$$

4 Proposed Algorithm

We extend the list scheduling algorithm ([2], [7], [8]) to solve the joint schedule problem. Each task in the task graph is assigned a priority value. Tasks are processed in decreasing order of their priorities. When processing a task, it is assigned to a selected resource so that it can complete as soon as possible. This is achieved by tentatively allocating this task to every compatible resource in the grid and comparing the outcomes of the different assignments.

The priority $P(T_i)$ for task T_i is determined by Eq. (7),

$$P(T_i) = \begin{cases} \varphi(T_i) & \text{if } T_i \text{ is an exit node,} \\ \max_{\forall j,(T_i,T_j)\in\mathcal{L}}\{P(T_j) + (\alpha_{(i,j)} & \text{otherwise.} \\ \quad +(1-\alpha_{(i,j)})\cdot\beta_{(i,j)})\cdot\varphi(T_j)\} & \end{cases} \tag{7}$$

Basically, $P(T_i)$ is an approximation of the least amount of time it takes to complete task T_i and all its offsprings in the task graph. $\varphi(T_i)$ is an approximate execution time of T_i:

$$\varphi(T_i) = \frac{\sum_{\forall h, t(M_h)=t(T_i)}}{\sum_{\forall h, t(M_h)=t(T_i)} p(M_h)} \cdot w(T_i). \tag{8}$$

If task T_j is assigned to a compatible resource M, the scheduling of T_j is determined as follows: (1) If T_j is an entry node, no constraint is imposed by its predecessors. Thus the earliest time when M is available is the time to start the execution of T_j. (2) If T_j is not an entry node, in addition to the resource contention, we need to inspect each of T_j's predecessors to determine the constraints imposed by the predecessors on the execution of T_j, by applying the analysis described in the previous section.

When a light-trail is needed for the data stream from one of T_j's predecessors, we consider using an existing light-trail and routing a new light-trail, whichever provisions the data stream earlier. When considering an existing light-trail, we need to make sure that it has enough spare capacity during the time interval the data stream sustains. Also, when we routing a new light-trail, all the wavelength-links on the trail should be available during the same time interval. Therefore we need to keep track of the status of every wavelength-link as well as every light-trail that has been established in the network, such that based on the status of the existing light-trails we can determine an earliest available light-trail that can support a connection or route a new light-trail.

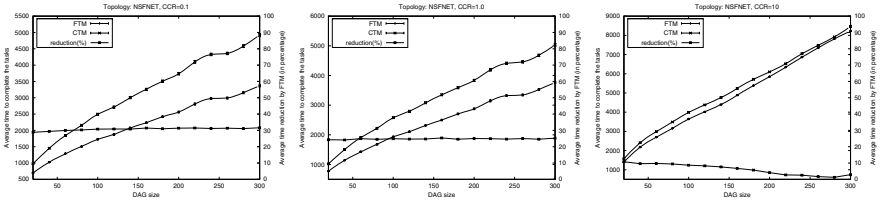
The details of the algorithms are not included in this paper due to space limit.

5 Performance Evaluation

We evaluate the performance of the proposed algorithm with three sets of randomly generated task graphs with different *communication-to-computing-ratio*s (CCR): 0.1, 1.0, and 10.0. CCR is defined as the average communication time

divided by the average execution time on a given system. The number of nodes in the DAGs (DAG size) varies from 20 to 300 with increments of 20.

Three topologies are used as the topologies of the WDM networks in our simulation: NSFNET, Torus 4 × 4, and a 16-node topology. We assume that the links in the topology are optical links with 4 wavelengths. The wavelength capacity C is assumed to be 1. Each node in the topologies is regarded as a switching node, to which a local cluster of resources are attached. 20 resources are randomly distributed in the grid. Each resource has processing power of 1 and is randomly assigned type τ_0 or τ_1.



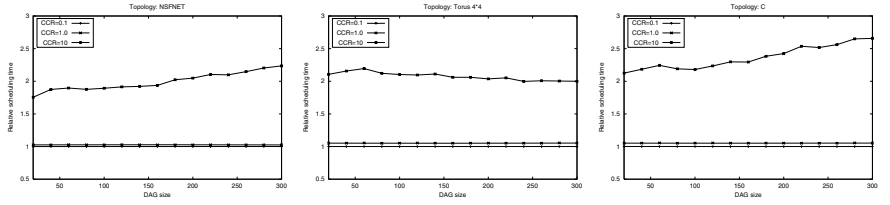
(a) (b) (c)

Fig. 4. Total task completion time on NSFNET

Figure 4 shows the simulation results for NSFNET (the results for the other two topologies are similar and are not included here). The curves indicated by FTM and CTM show the task complete time under FTM and CTM, respectively. The other curve indicates the percentage of the task completion time reduced under FTM. As we can see from the simulation results, under FTM the total task completion time is significantly reduced when compared with that under CTM. This is especially true for the cases where CCR=0.1 or 1, and the reduction of the task completion time ranges from 25% to 31% in all the three topologies. It shows FTM's excellent capability of capturing the concurrency characteristics between the tasks, as well as the efficiency of our scheduling algorithm. However, we notice that in all three topologies FTM and CTM have close performance when CCR=10, and the reduction of the task completion time ranges from 0 to 10%. This is reasonable since in this case the communication time is ten times the computation time. Thus, most of the time is consumed by transferring data between tasks such that the concurrency in the execution of tasks may not lead to significant reduction in the total task completion time.

At the same time, we observe that FTM uses more wavelength-links than CTM in most of the cases (figures not included due to space limit). This is because FTM models the output data of a task as a constant data stream which most likely results in subwavelength connections. Thus the optical channels in FTM are more likely to be under-utilized than in CTM.

We also compare FTM and FTM without communication constraints (FTM-NC). Figure 5(a), (b) and (c) present the task completion time under FTM in the three topologies relative to that of FTM-NC. The results show that in all the three topologies, when CCR=0.1 and 1.0, the communication overhead is very small, while for CCR=10 the task completion time of FTM is over 2 times that



(a) (b) (c)

Fig. 5. Total task completion time under FTM relative to that under FTM-NC

of FTM-NC. This shows that the communication service provisioning scheme is very efficient.

6 Conclusions

In this work a *flexible task model* (FTM) has been proposed for modeling the relationship between grid tasks. We investigated the problem of scheduling grid computing tasks modeled by FTM with light-trails in WDM networks to support the data communication between the tasks. Extensive simulations were conduced and results showed that our proposed task scheduling algorithm under FTM significantly reduced the total task completion time.

References

1. Garey, M., Johnson, D.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, New York (1979)
2. Wang, Y., Jin, Y., Guo, W., Sun, W., Hu, W., Wu, M.: Joint scheduling for optical grid applications. Journal of Optical Networking, 304–319 (March 2007)
3. Gumaste, A.: Light-trails as a SAN solution: Providing dynamic synchronous and multicasting connections in optical networks. SPIE Opticomm 2003, Workshop on Optical Networking Solutions for Global SAN (ONSAN), Dallas TX (Octobor 2003)
4. Gumaste, A., Chlamtac, I.: Adaptations to a GMPLS framework for IP over Optical Communication. National Fiber Optic Engineers Conferences (NFOEC), Orlando FL (September 2003)
5. Gumaste, A., Chlamtac, I.: Light-trails: A Novel Conceptual Framework for Conducting Optical Communications. In: Proceedings of IEEE Workshop on High Performance Switching and Routing, Torino Italy (June 2003)
6. Chlamtac, I., Gumaste, A.: Light-trails: A Solution to IP Centric Communication over Optical Networks. IEEE 2nd QoS IP Conference, Milan Italy (Februry 2003)
7. Topcuoglu, H., Hariri, S., Wu, M.Y.: Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Transactions on Parallel and Distributed Systems 13(3), 260–274 (2002)
8. Sinnen, O., Sousa, L.A.: List scheduling: extension for contention awareness and evaluation of node priorities for heterogeneous cluster architectures. IEEE Transactions on Parallel and Distributed Systems, 263–275 (March 2005)