OPTIMAL TASK-LEVEL SCHEDULING FOR CLOUD BASED MULTIMEDIA APPLICATIONS

Xiaoming Nan, Yifeng He, and Ling Guan

Department of Electrical and Computer Engineering, Ryerson University, Toronto, Canada

ABSTRACT

As an emerging computing paradigm, cloud computing has been increasingly used in multimedia applications. One fundamental challenge for application providers is how to effectively schedule multimedia tasks to multiple virtual machines for distributed processing. In this paper, we study task-level scheduling problem for cloud based multimedia applications. Specifically, we introduce a directed acyclic graph to model precedence constraints among tasks. Based on the model, we study the optimal task scheduling problem for the sequential, the parallel, and the mixed structures, respectively. Moreover, we propose a heuristic to perform the near optimal task scheduling in a practical way. Experimental results demonstrate that the proposed scheduling scheme can optimally assign tasks to virtual machines to minimize the execution time.

1. INTRODUCTION

Recent years have witnessed the fast development of cloud computing. In cloud data centers, a shared pool of servers are managed to provide on-demand computation, communication, and storage resources as accessible services. Owing to the elastic and on-demand natures of resource provisioning, cloud computing has been increasingly used in multimedia applications to effectively address intensive resource demands. In cloud based multimedia applications, application providers rent a certain number of virtual machines (VMs) to deliver multimedia services to users. For application providers, one major concern is how to optimally deploy multimedia tasks to diverse VMs for distributed processing. To address this problem, effective scheduling schemes are desired.

Generally, cloud computing employs two-level scheduling. The first level is the user-level scheduling, in which users' requests for one application are distributed to different VMs according to current workload. By balancing workload among VMs, the user-level scheduling can effectively avoid congestions in cloud. Compared to the user-level scheduling, the task-level scheduling is performed in a finer granularity. In general, a multimedia application can be decomposed into a set of tasks. Some tasks can run in parallel, while some tasks must be processed serially. The goal of task-level scheduling is to assign tasks to VMs so that the total execution time can be minimized. The two-level scheduling are performed in a hierarchical way in cloud. The optimal user-level scheduling have been studied in our previous work [1]. In this paper, we focus on the optimization of task-level scheduling for cloud based multimedia applications.

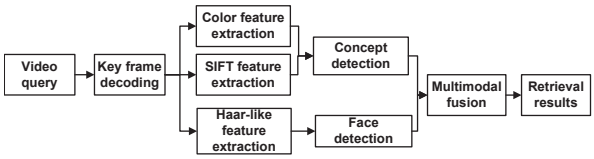

Fig. 1. An illustration of video retrieval framework.

In cloud, effective task-level scheduling can optimally match demands of tasks with resource provisioning of VMs and improve the Quality of Service (QoS). However, it is challenging to achieve the optimal task-level scheduling. Firstly, there are diverse precedence constraints, which are the precedence relations among tasks. For instance, Fig. 1 shows a video retrieval framework [2], in which the key frame decoding must precede feature extraction, and the concept detection cannot be executed until visual features are extracted. When assigning tasks to VMs, it is difficult to satisfy all these precedence constraints. Secondly, multimedia applications have different operation structures. Generally, there are three basic structures: *sequential*, *parallel*, and *mixed* structures. In the sequential structure, tasks are executed in a serial order, with one task starting after a previous task has completed. In the parallel structure, tasks can be performed concurrently. Mixed structure combines both sequential and parallel structures. It is a challenge to find an effective scheduling scheme for different structures. Thirdly, VMs have different resource capacities. It is challenging to assign a task to the best suitable VM. Finally, application providers require an efficient scheduling scheme, which can be adaptive to time-varying demands.

To address the above mentioned challenges, we propose an optimal task-level scheduling for cloud based multimedia applications in this paper. Our contributions can be presented as follows. We introduce a directed acyclic graph to characterize the precedence constraints among tasks. Based on the model, we optimize the task-level scheduling for the sequen-

tial, the parallel, and the mixed structures, respectively. In each structure, we formulate the task schedule optimization problem to minimize the total execution time. Since the formulated problem is NP-complete [11], we propose a heuristic to efficiently approach the optimal task-level scheduling.

2. RELATED WORK

Cloud based multimedia applications has attracted wide attentions from researchers. Zhu *et al.* [3] introduce the concept of multimedia cloud computing and propose the media edge cloud structure to reduce transmission delays. Nan *et al.* [4] propose queueing model based resource allocation to minimize the service response time. Wu *et al.* [5] integrate cloud servers with P2P streaming to support video on demand applications. Our work is different with topics in [3, 4, 5] and focuses instead on the optimization of task-level scheduling. Optimal scheduling in the distributed system has always been a challenging research topic. Tai *et al.* [6] propose a burst workload balancer to predict the changes in the user demands and accordingly shift schedule from greedy scheme to random scheme. Silberstein *et al* [7] present a schedule algorithm by adapting the multi-level queue approach in grid computing. But [6] and [7] only consider user-level scheduling. Tan *et al.* [8] study the optimization of subtask scheduling and communication traffic in a grid environment. Yu *et al.* [9] present a taxonomy for grid workflow system. However, [8] and [9] do not consider the constraint of resource cost, which is one major difference between cloud and grid computing [10]. Compared to previous work [5-9], our paper is different in following senses: 1) we optimize the task-level scheduling for cloud based multimedia applications; 2) we investigate task schedule optimization problem for the sequential, the parallel, and the mixed structures, respectively; 3) we propose a heuristic to efficiently approach the optimal task scheduling.

3. TASK SCHEDULING MODEL

In this section, we present our task scheduling model. To characterize precedence constraints among multimedia tasks, we introduce a directed acyclic graph (DAG), which is a directed graph with no path that starts and ends at the same vertex. The DAG can be denoted as: $DAG = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V}$ is the set of vertices and $\mathbb{E}$ is the set of edges. Suppose that a multimedia application can be decomposed into K tasks. Each vertex in DAG represents a task. Thus, $\mathbb{V} = \{V_1, V_2, \ldots, V_K\}$ represents the set of tasks. Each edge in DAG characterize a precedence constraint between two tasks. The edge $E_{k'k}$ represents that task V_k cannot be executed until task $V_{k'}$ has finished. Thus, the execution of task $V_{k'}$ is a precedence constraint for task V_k. Fig. 2 illustrates the DAG of the video retrieval framework in Fig. 1. There are 7 tasks in Fig. 2, and the video query and the retrieval result

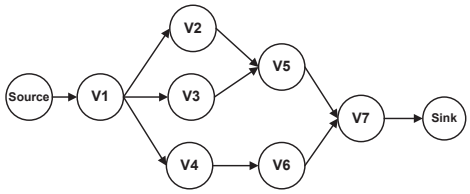

Fig. 2. The DAG of the video retrieval framework in Fig. 1.

are denoted as source and sink, which represent the start and the end of DAG.

Suppose that there are N types of VM instances. Each task can be served by any VM instance. But different types of VM instances have different price rates and resource capacities, leading to different execution time. Let t_j^k denote the execution time by using type-j VM instance to execute task V_k, and let p_j be the resource cost for renting one type-j VM instance. We assume that each task can only be assigned to one VM instance for execution. Therefore, the purpose of task-level scheduling is to optimally assign tasks to VMs. To represent the task assignment, we introduce s_j^k, which is given by

$$s_j^k = \begin{cases} 1 & \text{if } V_k \text{ is assigned to a type-}j \text{ VM instance,} \\ 0 & \text{otherwise.} \end{cases}$$

We will propose the optimal task schedule scheme to determine the optimal value of s_j^k $(V_k \in \mathbb{V}, \forall j = 1, 2, \ldots, N)$ to minimize the total execution time.

4. OPTIMAL TASK-LEVEL SCHEDULING

In this section, we use the model presented in Section 3 to study the optimal task-level scheduling for the sequential, the parallel, and the mixed structures, respectively. Moreover, we propose a heuristic to efficiently approach the optimal task scheduling.

4.1. Sequential Structure

We first investigate the sequential structure, in which all tasks are executed serially. Our objective is to minimize the total execution time. As described in Section 3, one task can only be assigned to one VM instance. If the task V_k is scheduled to type-j VM instance, $s_j^k = 1$ and $s_{j'}^k = 0$ $(j' \neq j)$. Thus, the execution time of V_k is given by $\sum_{j=1}^{N} s_j^k t_j^k$, which represents that the type-j VM instance takes t_j^k to execute V_k if V_k is scheduled to the VM instance. Since all tasks are executed serially in the sequential structure, the total execution time is the sum of the execution time for each task. Therefore, the total execution time is given by $T_{seq}^{tot} = \sum_{k=1}^{K} \sum_{j=1}^{N} s_j^k t_j^k$. The cost for processing V_k is denoted by $\sum_{j=1}^{N} s_j^k p_j$. Thus, the total resource cost for serving the multimedia application is

$C^{tot}_{seq} = \sum_{k=1}^{K} \sum_{j=1}^{N} s_j^k p_j$. In addition, every task should be assigned to one VM instance for execution. Thus, constraints $\sum_{j=1}^{N} s_j^k = 1$ ($V_k \in \mathbb{V}$) have to be satisfied. Based on the above analysis, we can formulate the task schedule optimization problem for the sequential structure as follows.

$$\begin{aligned} \underset{\{s_j^k\}}{\text{Minimize}} \quad & \sum_{k=1}^{K} \sum_{j=1}^{N} s_j^k t_j^k \\ \text{subject to} \quad & \\ & \sum_{k=1}^{K} \sum_{j=1}^{N} s_j^k p_j \le C_{upp}, \\ & \sum_{j=1}^{N} s_j^k = 1, \\ & s_j^k \in \{0,1\},\ V_k \in \mathbb{V},\ \forall j = 1, \ldots, N, \end{aligned} \tag{1}$$

where C_{upp} is the upper bound of resource cost. The optimization problem (1) is a 0-1 integer programming, which can be solved by enumerating every possible s_j^k. But the time complexity of enumeration cannot satisfy the real-time requirement for the practical application. Therefore, we propose a heuristic in Section 4.4 to efficiently solve problem (1).

4.2. Parallel Structure

In this subsection, we study the optimal task-level scheduling for the parallel structure, in which all tasks can be executed concurrently. Thus, the application can deploy different tasks to different VMs for distributed processing. According to Section 3, the execution time of V_k is given by $\sum_{j=1}^{N} s_j^k t_j^k$. The total execution time in the parallel structure depends on the largest execution time among all tasks, which can be formulated as $T^{tot}_{par} = \max_{\{V_k \in \mathbb{V}\}} \{\sum_{j=1}^{N} s_j^k t_j^k\}$. The total resource cost is given by $C^{tot}_{par} = \sum_{k=1}^{K} \sum_{j=1}^{N} s_j^k p_j$. Additionally, constraints $\sum_{j=1}^{N} s_j^k = 1$ ($V_k \in \mathbb{V}$) are required to guarantee all tasks are executed. Therefore, the task schedule optimization problem for the parallel structure can be formulated as

$$\begin{aligned} \underset{\{s_j^k\}}{\text{Minimize}} \quad & \max_{\{V_k \in \mathbb{V}\}} \{\sum_{j=1}^{N} s_j^k t_j^k\} \\ \text{subject to} \quad & \\ & \sum_{k=1}^{K} \sum_{j=1}^{N} s_j^k p_j \le C_{upp}, \\ & \sum_{j=1}^{N} s_j^k = 1, \\ & s_j^k \in \{0,1\},\ V_k \in \mathbb{V},\ \forall j = 1, \ldots, N. \end{aligned} \tag{2}$$

The optimization problem (2) is a 0-1 integer programming, which can be solved by the enumeration method.

4.3. Mixed Structure

We extend our study to the mixed structure in this subsection. In the mixed structure, some tasks can be processed in parallel, while some tasks must be executed serially. The video retrieval framework in Fig. 1 is an example of mixed structure. In the DAG of mixed structure, there are multiple paths from source to sink. The tasks in each path must be executed sequentially, while different paths can be taken as parallel structures. Suppose that there are W paths in a DAG, and let Ψ_w denote the set of vertices on the w^{th} path. The execution time of V_k is given by $\sum_{j=1}^{N} s_j^k t_j^k$, and thus the execution time for Ψ_w can be formulated as $\sum_{k \in \Psi_w} \sum_{j=1}^{N} s_j^k t_j^k$. Therefore, the total execution time is $T^{tot}_{mix} = \max_{\{w \in W\}} \left\{ \sum_{V_k \in \Psi_w} \sum_{j=1}^{N} s_j^k t_j^k \right\}$, which represents the largest execution time among all paths. The total resource cost is $C^{tot}_{mix} = \sum_{k=1}^{K} \sum_{j=1}^{N} s_j^k p_j$. Moreover, we need to satisfy constraints $\sum_{j=1}^{N} s_j^k = 1$ ($V_k \in \mathbb{V}$) to process all tasks. Therefore, we can formulate the optimal task schedule problem for the mixed structure as follows.

$$\begin{aligned} \underset{\{s_j^k\}}{\text{Minimize}} \quad & \max_{\{w \in W\}} \left\{ \sum_{V_k \in \Psi_w} \sum_{j=1}^{N} s_j^k t_j^k \right\} \\ \text{subject to} \quad & \\ & \sum_{k=1}^{K} \sum_{j=1}^{N} s_j^k p_j \le C_{upp}, \\ & \sum_{j=1}^{N} s_j^k = 1, \\ & s_j^k \in \{0,1\},\ V_k \in \mathbb{V}, \\ & \forall j = 1, 2, \ldots, N,\ \forall w = 1, 2, \ldots, W. \end{aligned} \tag{3}$$

The optimization problem (3) is a 0-1 integer programming.

4.4. Heuristic for Optimal Task Scheduling

The task schedule optimization problems (1), (2), and (3) are all 0-1 integer programming, which is NP-complete [11]. By exhaustively searching along both task dimension and VM dimension, the optimal solutions can be achieved. However, simply using the enumeration method is quite inefficient, especially when the number of tasks is huge. Therefore, we propose a heuristic to efficiently approach the optimal task scheduling. The core of the heuristic is to find critical tasks [12], which are on the longest path and determine the total execution time. By speeding up the execution of critical tasks, we can reduce the total execution time. The proposed heuristic is presented in Algorithm 1.

5. PERFORMANCE EVALUATION

In this section, we perform experiments to evaluate the proposed optimal task-level scheduling. Our experiments consist of the numerical simulation and the practical multimedia application.

We first verify the proposed task scheduling scheme in the numerical simulation. Amazon EC2 [13] is Amazon's cloud computing platform allowing application providers to rent VMs for services. To make our evaluation convincible, we use price rates and VM configurations of Amazon EC2 in our simulation. Four types of VMs are provided, including small, large, extra large standard instances, and medium high-CPU instances. The detailed configuration and price rate can be found from [13]. The number of tasks varies from 40 to 200 and the execution time of each task is Gaussian distributed with $(\mu = 1.5, \sigma^2 = 0.5)$.

Algorithm 1 Heuristic for Optimal Task Scheduling

1: Schedule each task to the cheapest VM and compute C^{tot} and T^{tot}. Let τ^k and C^k denote current execution time and resource cost for V_k, respectively.
2: If $C^{tot} > C^{upp}$, no solution exists.
3: **while** $C^{tot} < C^{upp}$, **do**
4: Calculate the earliest start time $e^k \leftarrow \max\{e^{k'} + \tau^{k'} | E_{k'k} \in \mathbb{E}\}$, and the latest start time $l^k \leftarrow \min\{l^{k''} - \tau^{k''} | E_{kk''} \in \mathbb{E}\}$. Moreover, initialize $e^1 \leftarrow 0$. If $e^k = l^k$, V_k is a critical task.
5: **for** each critical task V_k, **do**
6: If $t_j^k < \tau^k$, $\Delta t_j^k \leftarrow \tau^k - t_j^k$, $\Delta C_j^k \leftarrow p_j - C_k$, $\rho_j^k \leftarrow \frac{\Delta t_j^k}{\Delta C_j^k}$, which represents that reschedule V_k to type-j VM needs ΔC_j^k more cost but saves Δt_j^k time.
7: **end for**
8: Sort all ρ_j^k in descending order.
9: Select maximal ρ_j^k and reschedule V_k to type-j VM.
10: Update $T^{tot} \leftarrow T^{tot} - \Delta t_j^k$, $C^{tot} \leftarrow C^{tot} + \Delta C_j^k$, $\tau^k \leftarrow t_j^k$, $C^k \leftarrow p_j$.
11: **end while**
12: **return** T^{tot}.

In the simulation, we compare the total execution time among the proposed optimal task schedule scheme, in which tasks are scheduled by solving optimization problems (1), (2), and (3), the proposed heuristic in Algorithm 1, and the static execution scheme, in which all tasks are executed on the medium standard instance. The optimal task schedule scheme is achieved by enumeration, which is globally optimal benchmark but not efficient, while the proposed heuristic is near optimal but lightweight and practical. The comparison results of the total execution time for the sequential, the parallel, and the mixed structures are shown in Fig. 3(a), 3(b), and 3(c), respectively. From comparison results, we can see that the proposed optimal task schedule scheme can achieve lower execution time compared to the proposed heuristic and the static execution scheme under the same resource cost constraint. Additionally, we can find that the total execution time acquired by the proposed heuristic is quite close with the globally optimal solution. Thus, the proposed heuristic can approach the optimal task scheduling in an efficient way. The static execution scheme employs one type of VM instance to execute all tasks, thus the application cannot be effectively processed in a distributed manner, leading to a longer execution time. Comparing Fig. 3(a), 3(b), and 3(c), we also find that the total execution time in the mixed structure is longer than that in the parallel structure but lower than that in the sequential structure, which conforms Amdahl's law [14] that the speed-up of a program from parallelization is limited by the sequential portion of the program. Since the sequential structure has the largest sequential portion, the execution time for the sequential structure is the longest.

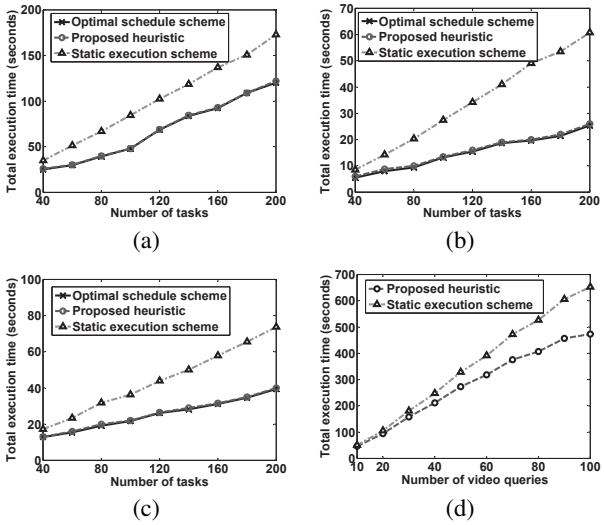

Fig. 3. Comparison results of the total execution time (a) for sequential structure, (b) for parallel structure, (c) for mixed structure, (d) for video retrieval framework.

Next, we evaluate the proposed heuristic for optimal task scheduling by applying it to the video retrieval framework [2]. All video queries and video database are from TRECVID 2009 search task [15]. Each video query is around 5 seconds. VMs are supported by two servers (Intel i7 CPU 3.07GHz, 8G RAM, and 1T hard drive). The comparison of the total execution time between the proposed heuristic and the static execution scheme is shown in Fig. 3(d). From Fig. 3(d), we can see that the proposed heuristic can process video queries with lower execution time than the static scheme. When the number of queries is 100, the proposed heuristic can reduce the total execution time by 28% compared to the static execution scheme.

6. CONCLUSION

In this paper, we study the optimal task-level scheduling for cloud based multimedia applications. We first introduce a directed acyclic graph to characterize the precedence constraints among tasks. Based on the model, we optimize the task scheduling for the sequential, the parallel, and the mixed structures, respectively. For each structure, we investigate the task schedule optimization problem to minimize the total execution time under the resource cost constraint. Moreover, we propose a heuristic to efficiently approach the optimal task scheduling in a practical way. Experimental results demonstrate that the proposed task-level scheduling scheme can effectively match demands of multimedia tasks with resource provisioning of VMs and achieve the minimal execution time for cloud based multimedia applications.

7. REFERENCES

[1] X. Nan, Y. He, and L. Guan, "Optimization of workload scheduling for multimedia cloud computing," Submit to *2013 IEEE International Symposium on Circuits and Systems*.

[2] Z. Zhao, Y. Zhao, Z. Gao, X. Nan, M. Mei, H. Zhang, H. Chen, X. Peng, Y. Chen, J. Guo, et al., "Bupt-mcprl at trecvid 2009," *TREC Video Retrieval Evaluation Online Proceedings, Gaithersburg, MD, USA*, 2009.

[3] W. Zhu, C. Luo, J. Wang, and S. Li, "Multimedia cloud computing," *IEEE Signal Processing Magazine*, vol. 28, no. 3, pp. 59–69, 2011.

[4] X. Nan, Y. He, and L. Guan, "Optimal resource allocation for multimedia cloud based on queuing model," in *Proc. IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2011.

[5] Y. Wu, C. Wu, B. Li, X. Qiu, and F. Lau, "Cloudmedia: When cloud on demand meets video on demand," in *Proc. IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2011, pp. 268–277.

[6] J. Tai, J. Zhang, J. Li, W. Meleis, and N. Mi, "Ara: Adaptive resource allocation for cloud computing environments under bursty workloads," in *Proc. IEEE Performance Computing and Communications Conference*, 2011.

[7] M. Silberstein, D. Geiger, A. Schuster, and M. Livny, "Scheduling mixed workloads in multi-grids: the grid execution hierarchy," in *Proc. IEEE Symposium on High Performance Distributed Computing*, 2006, pp. 291–302.

[8] M. Tan, H.J. Siegel, J.K. Antonio, and Y.A. Li, "Minimizing the application execution time through scheduling of subtasks and communication traffic in a heterogeneous computing system," *IEEE Transactions on Parallel and Distributed Systems*, vol. 8, no. 8, pp. 857–871, 1997.

[9] J. Yu and R. Buyya, "A taxonomy of scientific workflow systems for grid computing," *Sigmod Record*, vol. 34, no. 3, pp. 44–49, 2005.

[10] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Proc. IEEE Grid Computing Environments Workshop*, 2008.

[11] A. Schrijver, *Theory of linear and integer programming*, Wiley, 1998.

[12] J.E. Kelley, "The critical-path method: Resources planning and scheduling," *Journal of Industrial scheduling*, pp. 347–365, 1963.

[13] Amazon EC2, [online] Available: http://aws.amazon.com/ec2/.

[14] G.M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proc. ACM spring joint computer conference*, 1967, pp. 483–485.

[15] P. Over, G.M. Awad, J. Fiscus, M. Michel, A.F. Smeaton, and W. Kraaij, *TRECVID 2009-goals, tasks, data, evaluation mechanisms and metrics*, National Institute for Standards and Technology (NIST), 2010.