Fine-Grained Profiling for Data-Intensive Workflows

Nan Dun* Kenjiro Taura† Akinori Yonezawa‡
*‡*Department of Computer Science,* †*Department of Information and Communication Engineering*
Graduate School of Information Science and Technology, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656 Japan
**dunnan@yl.is.s.u-tokyo.ac.jp* †*tau@logos.ic.i.u-tokyo.ac.jp* ‡*yonezawa@is.s.u-tokyo.ac.jp*

***Abstract*—Profiling is an effective dynamic analysis approach to investigate complex applications. *ParaTrac* is a user-level profiler using file system and process tracing techniques for data-intensive workflow applications. In two respects ParaTrac helps users refine the orchestration of workflows. First, the profiles of I/O characteristics enable users to quickly identify bottlenecks of underlying I/O subsystems. Second, ParaTrac can exploit fine-grained data-processes interactions in workflow execution to help users understand, characterize, and manage realistic data-intensive workflows. Experiments on thoroughly profiling Montage workflow demonstrate that ParaTrac is scalable to tracing events of thousands of processes and effective in guiding fine-grained workflow scheduling or workflow management systems improvements.**

***Keywords*-profile, workflow, file system trace**

I. Introduction

Recent advances in the cluster, grid, and cloud computing enable users to execute various data-intensive workflows by harnessing widely available computing resources [1]. However, planning and scheduling the execution of complex workflows in distributed environments still remain challenging [2]. One of important demands is to understand and characterize the realistic behaviors of data-intensive workflows to help workflow management systems refine their orchestration for optimal workflow execution.

In response to this practical demand, research has been conducted to elaborate the characterization of a wide variety of scientific workflows using synthetic approaches [3], [4]. Though these methods are capable of understanding the basic structures of representative workflows from domain researchers, they show a lack of capturing fine-grained data-processes activities and it will be ideal if users can acquire the workflow essentials from their own workflows.

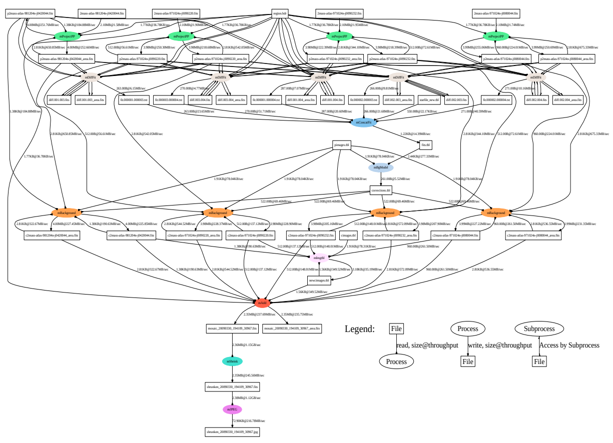
Profiling is an effective dynamic analysis approach to investigate complex applications in practice. Accordingly, we designed and implemented a user-level profiler *ParaTrac* for data-intensive workflow applications using file system and process tracing techniques. First, ParaTrac can produce valuable file system call statistics and input/output characterizations for I/O subsystem (e.g., distributed file system) performance analysis and tuning. Second, ParaTrac is novel because it provides realistic and comprehensive profiles of data-intensive workflows by exploiting fine-grained data-process interactions. These informative essentials enable users to understand and characterize the workflows, and therefore refine the planning, scheduling, and execution of complex workflows in distributed environments.

II. Profiling Approaches

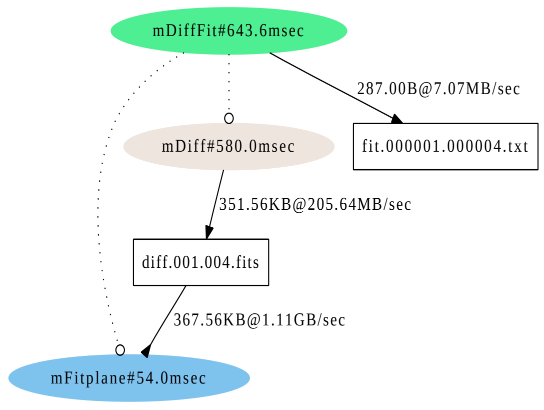
Tracing an application and producing profiles by ParaTrac are straightforward and effortless. First, using ParaTrac, user creates a traced file system via which application can be executed without modification simply by changing working directory or using `chroot` utility. User-level techniques used to trace both files and processes activities of unmodified executables includes FUSE (Filesystem in Userspace), `/proc` file system, and kernel process accounting.After the application finished, all system call events and process activities are stored into raw time-series logs for further processing. Then, the trace logs produced from individual trace instances are collected and integrated into one global SQL-queryable database. Finally, using the profile generation utility, the profiles of application are created by synthetically mining the trace data in database.

For I/O characterization profiles, statistical analysis is applied to reveal the overall behaviors of data manipulation, such as the variances of system calls and the I/O regularity and sequentiality of workloads.

For workflow analysis, ParaTrac uses casual and temporal analysis to generate process hierarchical tree (i.e. `fork` graph) as well as the workflow DAG (Directed Acyclic Graph) annotated with fine-grained runtime statistics to illustrate actual data-processes interactions in workflow execution. For workflows that are executed in distributed environments, the DAGs are aggregated from trace logs produced at each host. Then various manipulations can be applied to DAGs for further analysis. For example, the analysis of sub-workflow is available by partitioning DAG into subgraphs. The scheduling analysis can be achieved by aggregating processes-level DAG into job/task level DAG. In addition, graph-theoretic algorithms can be applied to node and edge attributes to explore specific tasks or data in workflow. For example, file nodes having the most degrees may indicate the critical node in workflow.



(a) Complete Workflow DAG



(b) Sub-DAG of Workflow

Figure 1. Fine-Grained DAG of Montage Workflow

III. Experiments

Our experiments use ParaTrac to profile the execution of Montage scientific workflow for different data sets on 8 Linux servers in the wide-area environments.

Experimental results show that ParaTrac introduces about 16% tracing overhead, which is mainly due to the context switch in FUSE and the tracing scalability of one trace instance mainly depends on the capability of FUSE handling concurrent requests. The size of tracing data primarily scales with the number of system call events, and is secondarily correlated with the number of processes born during the execution. For montage workflow, the trace data are approximately proportional (about 10%) to the size of the application data processed by the whole workflow.

From system call and I/O profiles, system calls with high latency in Montage workflow can be easily discovered. For example, operations that manipulate remote data having much higher latency than average ones indicate a proper data prefetch or replication strategy should be applied.

Figure 1 illustrates the remarkable feature of ParaTrac: the DAGs of Montage workflow with very detailed data-processes dependencies and interactions.

- *Process*: Processes (`name#lifetime` in oval) with their parent-child relationships (dot line with 'o' arrowhead) during the workflow execution.
- *Data*: Files (`filename` in box) accessed during the workflow execution are annotated with the size, portion and transfer rate (`size#rate` on edge) of data accessed by and passed among processes.

Comparing to the workflow DAGs by Pegasus Workflow-Generator [3], [5], ParaTrac provides more informative and fine-grained DAGs. Since more realistic runtime information can be obtained in details by profiling than synthetic analysis, we suggest that workflow management systems provide extra parameters specifying the resource constraints to allow users to perform the fine-grained scheduling of workflows. For example, a workflow management system can utilize the data pass volume and transfer rates obtained in workflow profiles to achieve a data throttling strategy to improve the throughput of workflows.

IV. Conclusion and Future Work

We have presented the design and implementation of ParaTrac — a profiler targeted for data-intensive workflows. Experiments on thoroughly profiling Montage workflow by ParaTrac demonstrate its scalability of tracing events of thousands of processes and its effectiveness by guiding fine-grained workflow scheduling and workflow management systems improvement.

In the future, we plan to expand ParaTrac to trace more workflow information, such as CPU time of tasks, to enrich the profiles of applications. Another direction is to reuse the profiles as a macro benchmark for workflow management systems by consistent replaying of profiles.

ParaTrac is an open source software and online available at http://paratrac.googlecode.com/.

References

[1] I. Taylor, E. Deelman, D. B. Gannon, and M. S. (Eds), *Workflows for e-Science*. Springer-Verlag, 2006.

[2] S. Pandey and R. Buyaa, "Scheduling and management techniques for data-intensive appliation workflows," in *Data Intensive Distributed Computing: Challenges and Solutions for Large-Scale Information Management*, T. Kosar, Ed. IGI Global, 2009.

[3] S. Bharathi, A. Chervenak, E. Deelman, G. Mehta, M.-H. Su, and K. Vahi, "Characterization of scientific workflows," in *Proc. The 3rd Workshop on Workflows in Support of Large-Scale Science*, 2008, pp. 1–10.

[4] L. Ramakrishnan and D. Ganno, "A survey of distributed workflow characteristics and resource requirements," Indiana University, Tech. Rep. TR671.

[5] Workflow generator. [Online]. Available: http://vtcpc.isi.edu/pegasus/index.php/WorkflowGenerator/