Reviews of ISPDC 2023 - DaphneSched: Scheduler for integrated data analysis pipeline

March 20, 2024

1 Reviewer 1

Decision: strong accept
Comments: DaphneSched: A Scheduler for Integrated Data Analysis Pipelines

- This work introduces the proposed design of DaphneSched, a task-based scheduler forIDA pipelines, which is a software infrastructure for integrated data analysis (IDA) pipelines. Then well known self-scheduling schemes are incorporated in the software and two experiments are used to evaluate the HPC performance.
- This is very important work and could be accepted. However, the following minor revisions must be made.
    1. This site in the footnote is not accessible. http://https//daphne-eu.eu/ Please replace it by some references on this project that must be mentioned in Section I of the paper and please remove the footnote
    2. The two problems used in the evaluation of the scheduling methods, in section IV, favor the static method because the tasks assigned in each scheduling step are of regularly changing loads. It is recommended to e.g. consider using several linear systems of randomly changing dimensions in order to create tasks of irregularly changing loads and they could be multiplexed.
    3. Several references, in the related works on loop self-scheduling, are outdated. In fact the well known self-scheduling methods (e.g. GSS, TSS, FSS, etc) have been well presented in more recent papers (e.g. [26] or books). It is understandable that the authors want to give credit to the first paper that proposed the methods. However, due to the limited space of a conference paper this should not be done by excluding more recent advances to the loop self-scheduling topic. Therefore, the authors should include in the bibliography recent works on loop scheduling schemes and their application to grid and cloud computing such as the following.
    K. Kyriakopoulos, et al, An Optimal Scheduling Scheme for Tiling in Distributed Systems, 2007 IEEE International Conference on Cluster Computing, Austin, TX, 17 Sept 2007, pp. 267-274.

A. T. Chronopoulos, et al Multi-Dimensional Dynamic Loop Scheduling Algorithms, 2007 IEEE International Conference on Cluster Computing, Austin, Texas, USA, pp. 241-248, 17-20 September 2007.
H., Yiming, et al. . "A hierarchical distributed loop self-scheduling scheme for cloud systems." In 2013 IEEE 12th International Symposium on Network Computing and Applications, pp. 7-10. IEEE, 2013.
S. Penmatsa, et al . "Implementation of distributed loop scheduling schemes on the teragrid." In 2007 IEEE International Parallel and Distributed Processing Symposium, pp. 1-8. IEEE, 2007.

.

2 Reviewer 2

Decision: borderline
Comments: Summary -¿ The paper describes DaphneSched, the scheduler of the Daphne system, an infrastructure for integrated data analysis pipelines. Daphne assumes heterogeneous systems, and expresses applications in the Daphne DSL, which then goes through an MLIR compilation chain and ends up in the runtime system, which manages work partitioning and assingment. DaphneSched uses the self-scheduling techniques available in the LB4OMP library, and multiple victim selection and work stealing techniques, for work partitioning and work assignment, employing multiple task queues at different levels of the system hierarchy. DaphneSched is evaluated with two applications, Connected Components and Linear Regression, on two dual-socket Intel processors, with multiple dynamic loop-scheduling techniques.

Strengths

- DaphneSched, as part of a complex system, integrates interesting dynamic scheduling techniques
- The scheduling techniques implemented in DaphneSched are extensively evaluated on multicore systems.

Weaknesses

- The most important weakness of the paper, in my opinion, is that the evaluation is limited to comparing the different techniques within DaphneSched, but DaphneSched itself is not evaluated against traditional runtimes. Since LB4OMP already implements those techniques, and the current evaluation of DaphneSched is limited to CPUs only, it would be nice to showcase and understand the differences between OpenMP and Daphne in the context of multicores. Daphne is a broader system, and integrates its own compilation chain, but how does it compare with OpenMP, which is currently also able to manage heterogeneous devices?
- The paper refers to tasks as the smallest unit of work, but later, when describing task partitioning, you discuss "the work (or task) partitioner that divides tasks into partitions". The terminology here is a bit confusing.

I would assume that here you refer to a task as a higher-level unit of work? Then in Section II, the paper also states that "When a worker is free and idle, it will attempt to obtain new work items". Aren't those work items tasks, i.e. the smallest units of work? Or are they chunks which are later further partitioned?

- MSFC is not defined anywhere. LB4OMP defines FSC. What does "M" stand for?
- I found the paper not to be very clear on the interplay of work assignment and work stealing. Part of my confusion comes from the phrasing in Section II, where you refer to both self-scheduling techniques and work-stealing/victim selection techniques as work assignment strategies. Aren't your dynamic loop scheduling techniques also offering some work partitioning?
- What is the role of the compiler in work partitioning? This is not discussed at all in the paper.
- How are partial results from tasks aggregated/reduced? Is this a task specified by the compiler? Where does the runtime come in?
- In your evaluation, you discuss the effect of lock contention, which is hindering the performance of MSFC. Indeed in runtime systems, locking and synchronization of workers on queues is one of the most important aspects for performance, yet the approach in DaphneSched seems to be quite naive. At the same time, even if you minimize synchronization overheads by replacing global locks with a different queue implementation, contention can still happen. Also, it appears that the problem here is the number of threads and the decreasing granularity of the tasks. Have you considered setting some kind of tunable thresholds?
- Is the plan for DaphneSched to offload the scheduling technique selection to the user?

I believe that many of my concerns could be addressed in a final version of the paper, however, I believe that the paper would benefit from a better evaluation.

3 Reviewer 3

Decision: accept
Comments: The paper presents a complex implementation system model for a wide variety of methods/domains including data management, high performance computing and machine learning. It also includes scheduling schemes with self-scheduling and work-stealing schemes for task partitioning and assignment.

The presented versatility of the method is illustrated with applications and implementations of two larger test cases and show an efficiency increase of 13% compared to other schedulers. The principal steps of the Integrated Data Analysis method include work partitioning and work assignment. The DAPHNE infrastructure exploits primarily data parallelism while functional parallelism as well as distributed memory support might be included in future work.

4 Reviewer 3

Decision: accept
Comments: The paper describes DaphneSched which is a task-based scheduler for DAPHNE, a new open-source integrated data analysis (IDA) platform. A distinct feature of DaphneSched is its versatile design, which includes eleven task partitioning and four task assignment techniques. DaphneSched's performance is evaluated on two applications: a product recommendation system and a linear regression model. The presented experiments were conducted on two multi-core platforms (with 20 and 56 cores, respectively).

The motivation for the works is as follows: since integrated data analysis pipelines incorporate complex data management (DM), high performance computing (HPC), and machine learning (ML) steps into a single environment, an efficient and versatile scheduler is necessary for efficient execution of IDAs' tasks. The scientific challenge lies in the diverse computing characteristics and demands of IDA subsystems. As the paper proposes and evaluates such a scheduler, the main contribution of the paper is both interesting and useful practically. The paper matches the ISPDC topics.

The writing quality and overall technical presentation is good. The title and abstract appropriately explains the contribution. The subsequent parts of the paper are consistent, the discussion presented is logical and convincing for readers. The paper is rather easy to read, the authors clearly explained the principles of the proposed solution and the essential implementation details. The references and related-works section are appropriate. I've not spotted language errors (but consider that I'm not an English native speaker).

The drawbacks of the paper is a somewhat limited experimental comparison (two server setups and two applications only). As DaphneSched lacks of support for distributed-memory, only the self-scheduling and work-stealing techniques are employed. The paper should answer whether it is a possibility to incorporate other techniques, e.g. intended for distributed systems.

Also, some mistakes and confusing statements were spotted:

- VEE is not explained, nor referenced,
- misleading symbols for factoring (FAC vs. FAC2(,
- MFSC vs. FSC: is this the same algorithm? Assumed yes, but it should be corrected,
- in abstract one can read: "ten task partitioning and three task assignment techniques", similarly in Section III: "twelve partitioning schemes", actually eleven schemes (STATIC, SS, MFSC, GSS, TFSS, FAC2, TFSS, FISS, VISS, PLS, PSS) and four task assignment techniques are listed and compared.

5 My strategy to address the comment comments

- Remove footnote about Daphne's website and replace it with reference [reviewer 1]
- Use recent references suggested by the reviewer to highlight the use of DLS techniques [reviewer 1]
- Adding one sentence about why not evaluating this work against other than DAPHNE in Section 4 (experimental evaluation and discussion) [reviewer 2]
- Shows the difference between FSC and MFSC in a sentence also FAC and FAC2 [reviewer 2 and 4]
- add a reference for the (vectorized execution engine) VEE
- fix the count of the DLS schemes twelve should be eleven (MSTATIC is never mentioned in the paper)